TOWARDS THE FIFTH GENERATION

EUROPEAN MARKET OPPORTUNITIES 1988 - 1993

INPUT

About INPUT

INPUT provides planning information, analysis, and recommendations to managers and executives in the information processing industries. Through market research, technology forecasting, and competitive analysis, INPUT supports client management in making informed decisions.

Continuous-information advisory services, proprietary research/consulting, merger/acquisition assistance, and multiclient studies are provided to users and vendors of information systems and services (software, processing services, turnkey systems, systems integration, professional services, communications, systems/software maintenance and support).

Many of INPUT's professional staff members have more than 20 years' experience in their areas of specialization. Most have held senior management positions in operations, marketing, or planning. This expertise enables INPUT to supply practical solutions to complex business problems.

Formed as a privately held corporation in 1974, INPUT has become a leading international research and consulting firm. Clients include more than 100 of the world's largest and most technically advanced companies.

INPUT OFFICES

North America

Headquarters
1280 Villa Street
Mountain View, CA 94041
(415) 961-3300
Telex 171407 Fax (415) 961-3966

New York
Parsippany Place Corp. Center
Suite 201
959 Route 46 East
Parsippany, NJ 07054
(201) 299-6999
Telex 134630 Fax (201) 263-8341

Washington, D.C.
8298 C, Old Courthouse Rd.
Vienna, VA 22180
(703) 847-6870 Fax (703) 847-6872

International

Europe
INPUT LTD.
Piccadilly House
33/37 Regent Street
London SW1Y 4NF, England
01-493-9335
Telex 27113 Fax 01-629-0179

INPUT s.a.r.l.
29 rue de Leningrad
75008 Paris, France
01-44-80-48-43
Fax 01-44-80-40-23

Japan
FKI, Future Knowledge Institute
Saida Building,
4-6, Kanda Sakuma-cho
Chiyoda-ku,
Tokyo 101, Japan
03-864-4026 Fax 001-03-864-4114

DECEMBER 1988

TOWARDS THE FIFTH GENERATION

TOOLS AND RULES, 1988-1993

INPUT®

Piccadilly House, 33/37 Regent Street, London SW1Y 4NF
01-493-9335

Researched by
INPUT, LTD.
Piccadilly House
33/37 Regent Street,
London SW1Y 4NF
England

Published by
INPUT
1280 Villa Street
Mountain View, CA 94041-1194
U.S.A.

Software and Services Programme—Europe (SSPE)

Towards the Fifth Generation—Tools and Rules, 1988-1993

SFGE • 600 • 1988

Abstract

This report addresses a number of key questions concerning the methods and tools for developing software. These methods and tools are of vital importance to not only the software products vendors but to professional services firms as well. The report addresses such areas as:

- Fifth-generation software languages
- AI (Artificial Intelligence)
- KBS (Knowledge-Based Systems)
- CASE (Computer-Aided Software Engineering)

Other questions addressed are:

- Where does the fourth generation end?
- What is the future of third-generation languages such as COBOL?

The report examines the facts behind the mythologies of current system development techniques. It researches the trends, issues, and opportunities for all types of vendors in the dynamic $2.5 billion sector for application development products and services in Western Europe. Different solution approaches are assessed and placed in context in the overall market.

Development opportunities and marketing strategies in each country market are outlined. Particular emphasis is placed, in research and recommendations, on the need for tools, products and services to be integrated within the overall system life cycle. Formal methodologies, adopted within an Open Architecture environment, are recommended as part of this master framework for system development.

The report contains 150 pages, including 51 exhibits.

Table of Contents

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Exhibits

Exhibits (Continued)

Exhibits (Continued)

I

Introduction

I

Introduction

A

Objectives of the Report

This report has been written as part of INPUT's Software and Services Programme—Europe. Other areas covered during 1988 include application software products, network services, education and training, and systems integration. Included is the Annual European Industry Report, which will focus on the quantitative aspects of market size, forward forecasts and industry structure.

This report follows INPUT's research report issued as part of the 1987 programme, entitled *Computer-Aided Software Engineering in Europe, 1987-1992*. That report stressed the comprehensive nature of the concepts and methods that indicate a need for a total commitment when adopting this new technology.

The newness of the technology lies not so much in any breakthroughs on the part of individual techniques, but rather in the all-embracing nature of its demands.

The objectives of this second study and report are:

- To study the methodological framework within which the new CASE products and the techniques that underlie them must co-exist

- To understand the interactions between the traditional software production aids (languages, utilities, environments) and products that appear to be offering some new capability

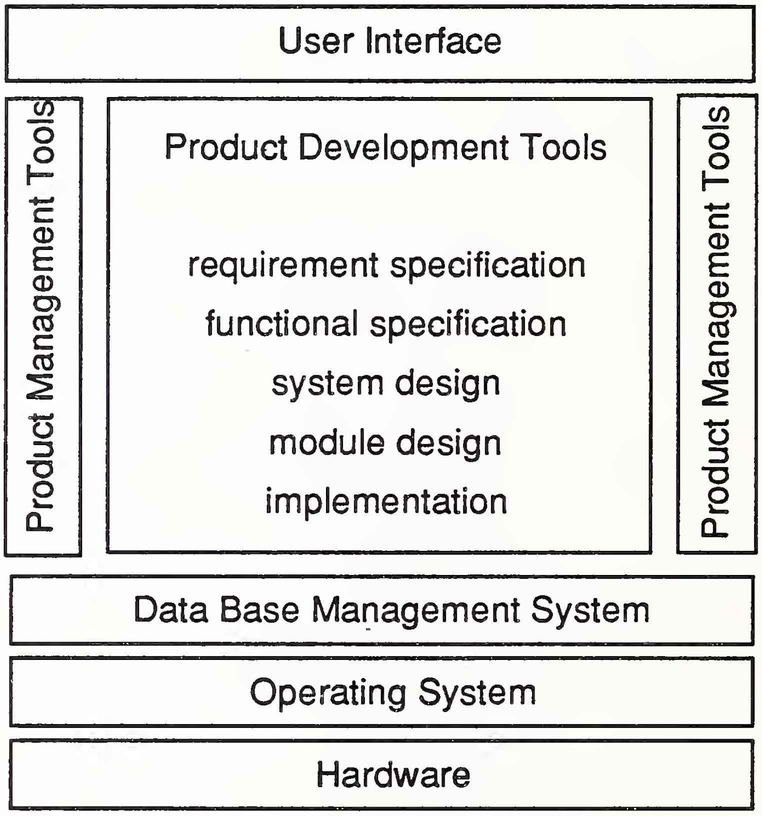
- To advise vendors in the different product and service areas of software on how to pursue their individual marketing programmes to maximise their companies' potential in the short and long term

B
Scope

Exhibit I-1 illustrates the elements of the Software Engineering (SE) Environment, which are increasing in complexity along with the size and complexity of the systems they are expected to construct. All of these elements are important in the production of systems, but if any one stands out from the rest it is the need for effective project management enhanced by appropriate project management tools.

EXHIBIT I-1

ELEMENTS OF SOFTWARE ENGINEERING ENVIRONMENTS



Most of the elements illustrated have been for many years recognised as part of the overall software products market-place:

- Operating system software

- Implementation aids in the form of editors, language processors, debuggers, test harnesses and test data generators

- File and database handlers

- Etc.

The current report aims to treat the whole market for products and services as one sector, which will include components of:

- Software products

- Professional services, such as education and training, consultancy and standard or tailored implementation services

- Turnkey systems, implementing complete in-house CASE systems

Hardware opportunities resulting from pull-through market forces in the services sectors are excluded from the main scope of this report (although touched on in general terms in the market overview section).

The geographic scope of the report covers principally the four main country markets of Western Europe (West Germany, France, U.K. and Italy), with some consideration of Scandanavia, Benelux and the rest of Europe.

C
Methodology

Field research for this report consisted of two interview programmes conducted during 1988:

- User research was conducted during February 1988 as part of INPUT's ongoing annual user research programme. One hundred user interviews were conducted with data processing managements across Europe. The questionnaire used appears as Appendix D.

- Vendor research was conducted from February until September 1988. This included primarily face-to-face and a minority of telephone interviews with hardware suppliers (offering and using SE products and methods), CASE tool vendors, consultancies specialising in SE methods, professional services vendors (using methods and tools for their own internal work and offering tools and consulting services on the open market), as well as research institutes and industry associations. Thirty-five vendors were interviewed in-depth, and in some cases more than one person was contacted. The vendor questionnaire appears in Appendix C.

Other public-domain sources—including company product literature, company annual reports and press releases, and technical and industry press articles—were read to obtain background data on market developments in Europe and the U.S.A.

Where country market statistics have been shown, local currencies have been converted to U.S. dollars on the basis of the average exchange rates for 1988, which are illustrated in Exhibit III-6. Owing to the volatility of international exchange rates anticipated for the period 1998 to 1993, INPUT has not attempted to forecast future rates. Currency conversion for this period has been made at the average 1988 rates, giving the forecasts in constant 1988 U.S. dollars.

D

Report Structure

The remaining chapters of this report are organised in the following manner:

- Chapter II is an Executive Overview that summarises contents of the report.

- Chapter III includes an overview of the market sector for Application Development Tools (ADTs). Issues and trends impacting the market are discussed, together with an assessment of market size and expected growth by major subsector.

- Chapter IV reviews the sector for methods and methodologies for systems engineering, including the aspects of tools, training, consultancy and other services.

- Chapter V reviews the sector for application tools, concentrating on the mutual interactions between different solutions embodied in different product groups: 4GLs, RDBMS, AI toolsets and CASE products.

- Chapter VI analyses the impact of CASE and its associated methodologies on the user community, including: manufacturers; Professional Service vendors; and commercial, industrial and military in-house users. This chapter presents the results of the survey of user experience and attitudes.

- Chapter VII reviews the technology streams impacting the application development tool sector, giving examples of work in the development stage in both Europe and the U.S.A.

- Chapter VIII presents INPUT's conclusions and recommendations for the short-term opportunities that are becoming available for each type of player in the sector.

- The Appendixes contain a list of definitions relevant to the subject matter of the report, an analysis of the research sample and the vendor and user questionnaires.

II

Executive Overview

II

Executive Overview

A

The Software Crisis

One of the main concerns of the IT industry has been that software is not being produced fast enough. This situation contrasts markedly with developments on the hardware side, where new and exciting possibilities appear regularly. Software on the other hand proliferates but appears to become no easier to provide quickly.

This problem manifests itself in an ever-growing applications backlog—measured at between 18 and 24 months in many DP shops and evidenced this year by INPUT's user research, where it was found that 42% of users had increased their backlogs, while only 21% had decreased them.

The other important worry is that software cannot be guaranteed to be reliable, to be bug-free, to be up to specification or even to be specifiable. This concern expresses itself in the use of the phrase *poor software quality*, which can be variously defined as:

- Software a user will not accept
- Software with an unacceptable level of bugs
- Software that cannot be understood from its documentation
- Software that degenerates as it is maintained
- Software that has to be rewritten because no one would risk enhancing it further

In face of the increasing need for software, estimates of the gap between the number of DP professionals required and the number being trained make the use of the phrase *software crisis* no exaggeration.

Exhibit II-1 lists the principal components of this crisis:

- Low productivity
- Poor quality of products
- Mounting backlog
- Chronic lack of expertise
- Project failures

EXHIBIT II-1

THE SOFTWARE CRISIS

- DP Productivity
- System Quality
- Containing the Backlog
  - Maintenance—50% to 70% of Effort
- Skills Shortage Worsening
- Search for On-Time, On-Cost Projects
  - No Failures

B Some Solutions

The objectives of this report are:

- To review the spectrum of application development methods and of computer-based development aids (which have been increasingly marketed either singly or in consort as a means of attacking the software crisis)

- To make recommendations to vendors as to how they should position their offerings in a confusing environment in which several approaches and many solutions appear to be offering salvation, often in ways diametrically opposed

Fourth-Generation Languages (4GLs) and their associated Database Management Systems have been hailed as the best answer to the productivity problem. In many DP shops producing standard commercial application suites, these languages have now found an established position alongside the older 3GLs, although not necessarily to the extent that would oust the previous generation altogether.

There are now at least 100 4GLs, which can be classified according to the type of user and the type of application to which they are best suited. There are, however, no official standards for these tools nor any agreed international definition as to what constitutes a 4GL. Moreover, 4GLs are now accused of having aggravated the programming skills shortage by needing and breeding a host of specialists who cannot move from one 4GL to another.

Structured methodologies have been around since the start of the 1970s. Recent moves to automate them, by providing on-line support tools to assist in many of the development stages, have brought them within the definition of the CASE tools market. Used in conjunction with the correct tools, structured methodologies offer the hope of providing the optimum solution to the software quality problem.

Artificial Intelligence (AI) is present in the market both as a sector in its own right, pursuing its own appropriate applications, and as an enabling technology supporting many of the objectives of the Software Engineering sector.

Exhibit II-2 lists the main components of the market's response to the crisis:

- Advanced languages
- Methodologies
- AI techniques
- Software engineering practice

C

Application Development Software Market

The overall application development software market for products and associated services (maintenance, education, training, consultancy and implementation services) in Western Europe for 1987 was $1,965 million, and INPUT forecasts that it will grow from this base to $7,900 million by 1993. By 1993 the products portion will be the biggest sector in the $25,000 million software products market.

The current growth rate of this market is 28% (between 1987 and 1988), but key segments are growing much faster—the front-end tools segment, for example, grew 75% for the year.

Over the 5-year forward period, the overall growth rate of the sector will be 26%, with products growing 24% per annum on average and services by as much as 36% per annum.

EXHIBIT II-2

SOME SOLUTIONS

- 4GLs Target Productivity (1)
  - But Accused Proliferation of Unique Systems
- Methodologies Target Quality (2)
  - But Accused 'Cumbersome' or 'Bureaucratic'
- AI Targets Complexity (3) and Flexibiity (4)
  - But Accused 'Academic Hype' or 'Irrelevant'
- Software Engineering
(Information Engineering, CASE)
Targets (1) to (4)
But "Cultural Migration" Expensive
  - 'Beyond the Flavour of the Month'

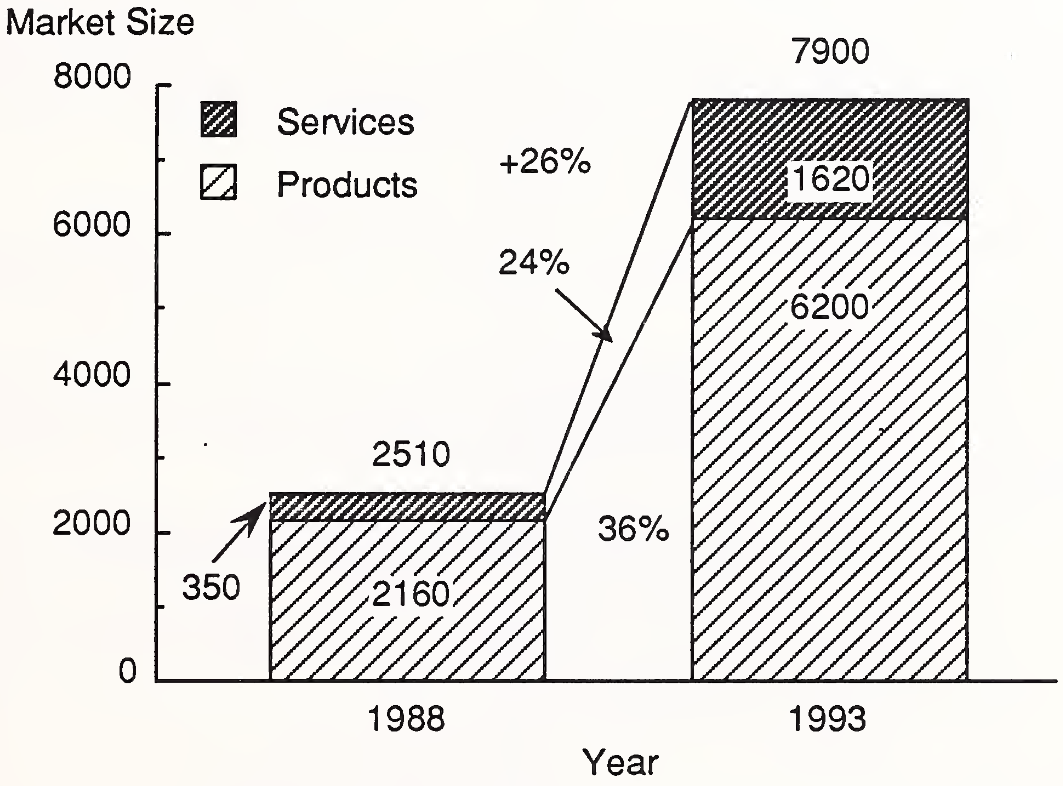
The principal driving forces fuelling this growth are:

- The gap between the demand for software and the supply of skilled professionals to produce it
- The growing size and complexity of major software projects in the systems integration business
- The increasing degree of reliance upon software in both commercial and industrial systems
- The safety-critical nature of major industrial systems
- The growing realisation that professionals must be given the quality tools to finish the job

Exhibit II-3 shows the overall growth of the market during the 5-year forward period between 1988 and 1993.

EXHIBIT II-3

APPLICATION DEVELOPMENT SOFTWARE— FORECAST REVENUES IN 1988 CURRENT $ MILLIONS



D

Forecasts for the Product/Service Mix

The 1988 market in Western Europe for application development software products measured $2,160 million, was growing at a rate of 27% over 1987 and will reach over $6,000 million by 1993. In addition, a software maintenance and support component will grow from $254 million in 1988 to over $800 million by 1993, as the installed base of products increases.

This market sizing and the forecasts include the software portion of standard turnkey systems sold in the I-CASE segment.

The major trends affecting the products segment are:

- Integration of new products and enhanced versions of older products into the life cycle

- Development of standard interfaces to allow tools to be integrated and to interconnect

- Development of interfacing components to allow unintegrated or incompatible tools to work together

- Separation of the development ('source') machine environment from the production ('target') machine environment in commercial DP shops

There is also a major professional pervices market sector opening up to service the user base's requirement for:

- Training in the use of methods and tools

- Implementation and consultancy services associated with their installation.

INPUT forecasts that the combined sectors for these two will grow from $96 million in 1988 to $780 million by 1993. In other words, in five years' time this service sector alone will account for over $750 million.

The cause of this phenomenal growth is that the difficulties of migrating from the present manual or "non-CASE" way of working for the many users who find themselves in the new-development- vs.-maintenance bind will present a multitude of service opportunities to the computing services companies.

Exhibit II-4 charts the 1988 and 1993 values of:

- The five product group segments in this market
- The three service sectors

E

The User Perspective

Twice as many users report an increase (as opposed to a decrease) in their backlogs, with just over a third maintaining the same level as previously (last measured as between 18 and 24 months).

Seventy-nine percent were in the position of having not decreased their backlogs during the year, down 7% from 86% year ago.

A majority (75%) of users use structured methodologies.

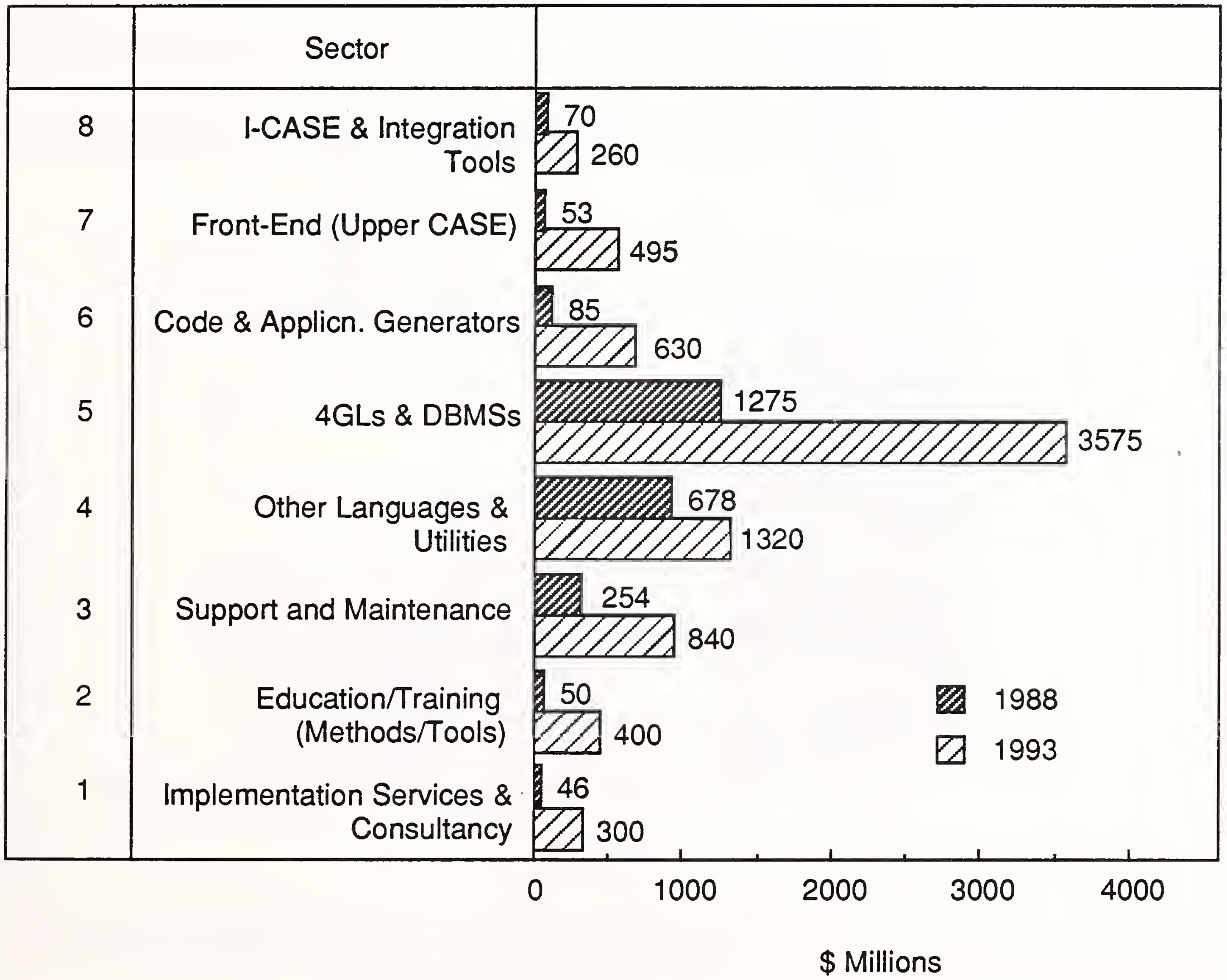
A minority (23%) use or are planning to use CASE products.

The commonest reasons for purchasing CASE tools are:

- Improved speed of development/productivity
- Improved quality using a standard approach

EXHIBIT II-4

APPLICATION DEVELOPMENT SOFTWARE PRODUCT/SERVICES, 1988-1993



- Flexibility and ease of use

Major difficulties encountered in evaluating CASE are:

- Adapting it to the user's needs
- Performance problems in high-volume transaction systems
- Deciding objectives

Of the 15% of users who already have some experience using CASE, the major implementation problems have been:

- Training and familiarisation
- Interest on the part of development staff
- Interfacing problems

Reasons for rejecting CASE include a great deal of ignorance—20% of users admitted ignorance.

Exhibit II-5 lists the major findings of INPUT's 1988 research into this topic.

EXHIBIT II-5

THE USER PERSPECTIVE

- The Backlog
- 75% Use Structured Methodologies
- 23% Use or Are Planning to Use CASE
- 15% Use CASE as Well as a Methodology
- 8% Planning to Use CASE
- Little Perception of Systems as Competitive Edge
- Causes of Rejection

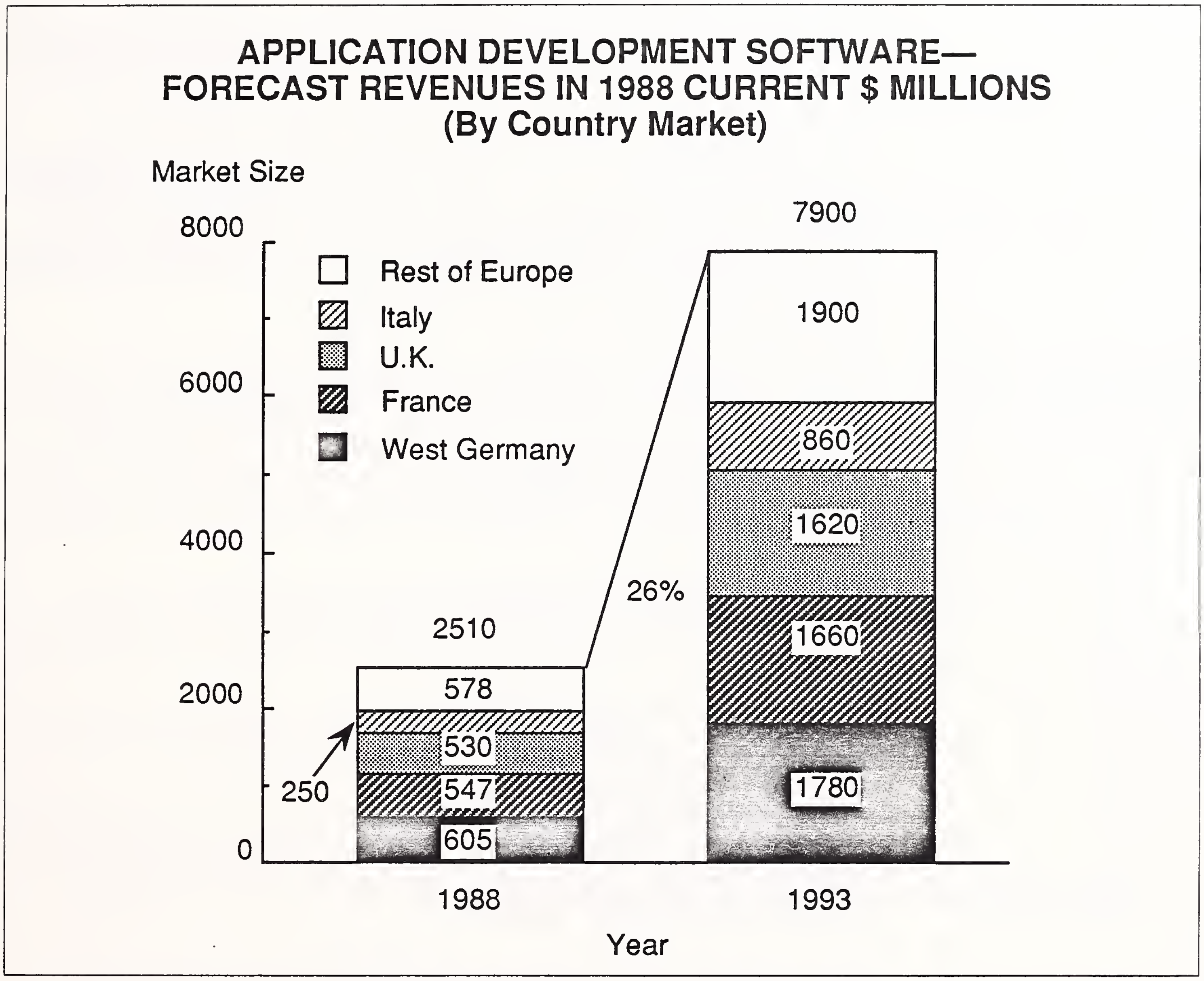
F The West European Market by Country

Exhibit II-6 shows the INPUT market sizing and forecasts for the software engineering sector by major country markets in Western Europe.

West Germany is the largest market and will retain this leadership over the forecast period.

In France, the tools market is seen as still small and immature. However, it is expected to catch up with West Germany and the U.K. during the forward 5-year period.

EXHIBIT II-6



The U.K. (followed by the rest of Europe) is the fastest growing market in Europe at the present time, particularly for front-end tools based on the analyst workbench, such as Arthur Young's IEW, and at the back end with code and application generators such as Cortex's CorVision and Pansophic's Telon.

The market in Italy will grow faster than in the other major countries but is starting from a relatively small base.

In Italy there are only a few relatively isolated islands of CASE competence in the large organisations. The penetration of structured methodologies is also low.

Scandinavia is deeply penetrated with the application of structured methodologies, and training will remain an important service sector.

Scandinavia and the Benelux countries are well-attuned to the need for CASE, and despite the market sizes are well penetrated with product.

In looking at other markets outside Europe, INPUT finds that in the U.S.A. there is a growing awareness of the need to structure the application of CASE within the framework of proven methodologies. In the Japanese market, there is a large shortfall in the number of computing graduates being produced by the education system and this is expected to fuel the need for advanced CASE systems

G

Key Trends and Recommendations

INPUT sees the present position in application development systems as analagous to what happened in the CAD/CAM market when people started to see the need to integrate CAD with CAM. The next five years is going to see this integration take place in the CASE market-place, and at the same time the overall framework (analagous to CIM) will start to be worked out.

The system life cycle is now being defined to cover the full circle of operations beyond maintenance to include enhancement and renewal of systems.

Over the next two years standards will evolve and IBM will sanction the technology by launching its own repository, based on DB2.

Loose coupling of methods to tools and vice versa will be achieved using meta-methodologies that allow for the tailoring of tools and methods to individual projects or to each DP shop—for proponents of loose coupling, ‘the methodology is a tool ‘.

Close coupling of tools to methods will be favoured by some vendors, e.g., James Martin Associates with the IEM. For proponents of close coupling, ‘the methodology is a rule ‘.

Vendors are recommended to consider three elements in the product planning of their marketing mix. These correspond to three waves of innovation that are currently overlapping in their arrival in the market-place:

- Training in methodologies and products
- Software tools addressed to individual life cycle functions
- Provision of complete systems—incorporating methods and tool kits and addressing the whole system life cycle

This overall life cycle must be considered when planning the evolution of product catalogues—in order to take into account new platforms, new architectures and changing software technology.

Exhibit II-7 lists the key trends and opportunities.

EXHIBIT II-7

KEY TRENDS AND RECOMMENDATIONS

- Full Life Cycle Approach
- Re-Engineering Key
- Market Check 18 Months Away
- Niche Segments vs. Full-Service Orientation
- Major Service Options in Longer-Term
- Opportunity Areas
  - Training
  - Migration Audits
  - Gaps in Life Cycle Coverage
  - Standard Turnkey Systems

III

The Market for Software Production Tools and Methods

III

The Markets for Software Production Tools and Methods

A
Market Overview

Software production has increasingly failed to keep up with the demand for new and complex systems. Whereas hardware capabilities have risen during the 1980s at a breathtaking pace, software has more and more been perceived as the industry bottleneck.

- In addition, as computer-based systems become ever more widespread, the safety factor (safety, that is, to human life and to commercial viability) becomes more and more critical.

Software engineering, conceived in the 1960s as the application of engineering principles to the production of software, has had a mixed reception in the market-place:

- In academic circles, software engineering is accepted virtually universally, with the exception of a few divergent voices from the social sciences discipline who question the ability to define certain elements in the equation and see this lack as the main drawback to the rigorous application of the scientific approach.

- In government and military procurement circles, the search for more controlled and standardised software production is underway, and the adoption of standards for design, testing, project management etc. on the part of contractors will become even more necessary than it is today.

- Among commercial DP management, the wish to produce quality software on time exists, but the ability to change technological horses in midstream is seen as the main stumbling block to the adoption of software engineering principles (CASE products have until recently been perceived as not cost-effective).

- Among the managers responsible for the production of industrial, real-time systems (and INPUT includes the hardware suppliers and systems houses here as well as in-house software developers), there has been greatest acceptance of the principles of software engineering and more attempts to put these principles into practice.

However, problems both practical and philosophical have arisen in the application of tools and engineering methods to software production. These problems result in the market-place of 1988 looking like a rag-bag of different approaches, all contending to be the true solution to the so-called software crisis.

A key question is:

- Are we trying to automate the process and thus do away with the intervention of software professionals, or are we trying to give them better tools to do the job?

- This isn't just an academic question, because the main way to help professionals is to give them computer assistance for routine work, and conversely, every time one automates a process, one requires a higher calibre person to manage the automated environment.

INPUT detects two different aims being addressed by suppliers in this sector:

- The incremental approach attacks the problem of low software productivity, with tools such as 4GLs and application generators in the vanguard of this battle.

- The radical approach attacks the problem of software quality, variously defined as "software the users will accept", "software that is understandable and maintainable" or "safe or robust software". Methodologies and total project support environments (IPSEs or I-CASE tools) are being rolled out against the enemy of low quality.

Several vendors claim their products achieve both aims simultaneously.

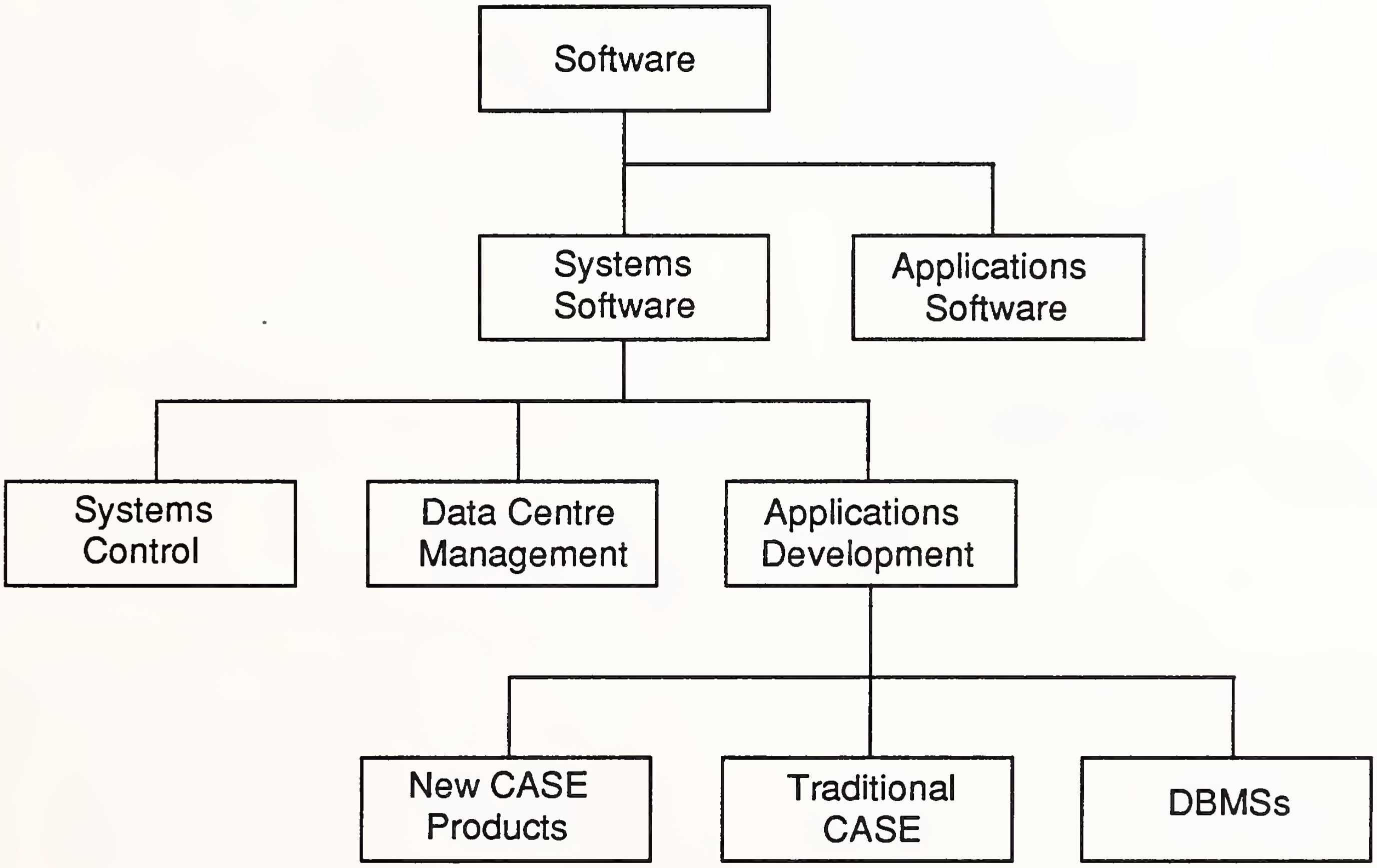
Users need to be able to establish a trade-off between these two approaches. The smaller the user, the greater the pressure to disregard anything except the short-term needs, hence extensive application of the incremental approach. The larger the user, the greater the need to adopt a radical approach to maintain control of large in-house IS departments.

In order to clarify the issues and to measure the market size and growth rates, this report surveys the current state of the software production market, defined to cover both products and services to assist in software production.

Exhibit III-1 shows a simplified structure of the software products market as defined by INPUT.

EXHIBIT III-1

STRUCTURE OF THE SOFTWARE PRODUCTS MARKET



Application Development Tools (ADTs), which form one of the three branches of Systems Software, is itself divided into sub-branches. These have been redefined for the purposes of this report to cover three areas:

- New CASE tools, defined as all products marketed as CASE tools or tool sets since the term CASE was introduced in the early 1980s

- Traditional CASE tools, otherwise defined as program development and production tools. These are included because all such products are a form of computer assistance to the software engineer, not just products that give themselves a CASE label.

- Database Management Systems (DBMSs), products that give an environment for any application that invokes their facilities, but have not been transferred to the System Control software category because they are application-enabling

Exhibits III-2 and III-3 show the breakdown of the New CASE tools sector, and of the rest of the market, including the traditional CASE tools sector. Last year's report, *Computer Aided Software Engineering in Western Europe*, included INPUT's 1987 market sizing and forecasts for the new CASE tools sector only (although it was not called that in 1987). This year INPUT is extending its market sizing and forecasting to include both new and traditional sectors, as well as commenting on the methodologies sector. The basis is that all ADTs must now be used and seen to be used within the field of software production as an engineering discipline. The trend is towards this type of integration within an accepted framework known as the system life cycle.

EXHIBIT III-2

STRUCTURE OF THE NEW CASE TOOLS MARKET

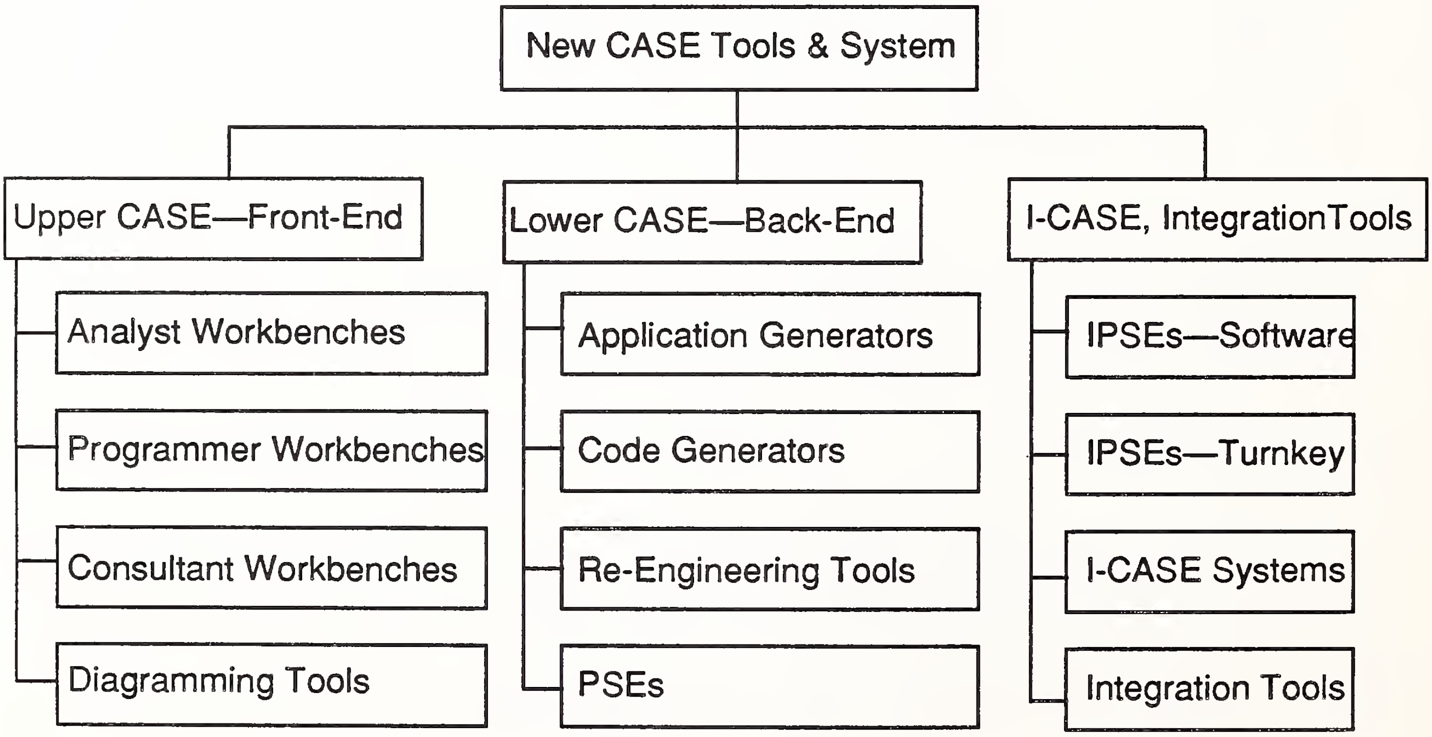
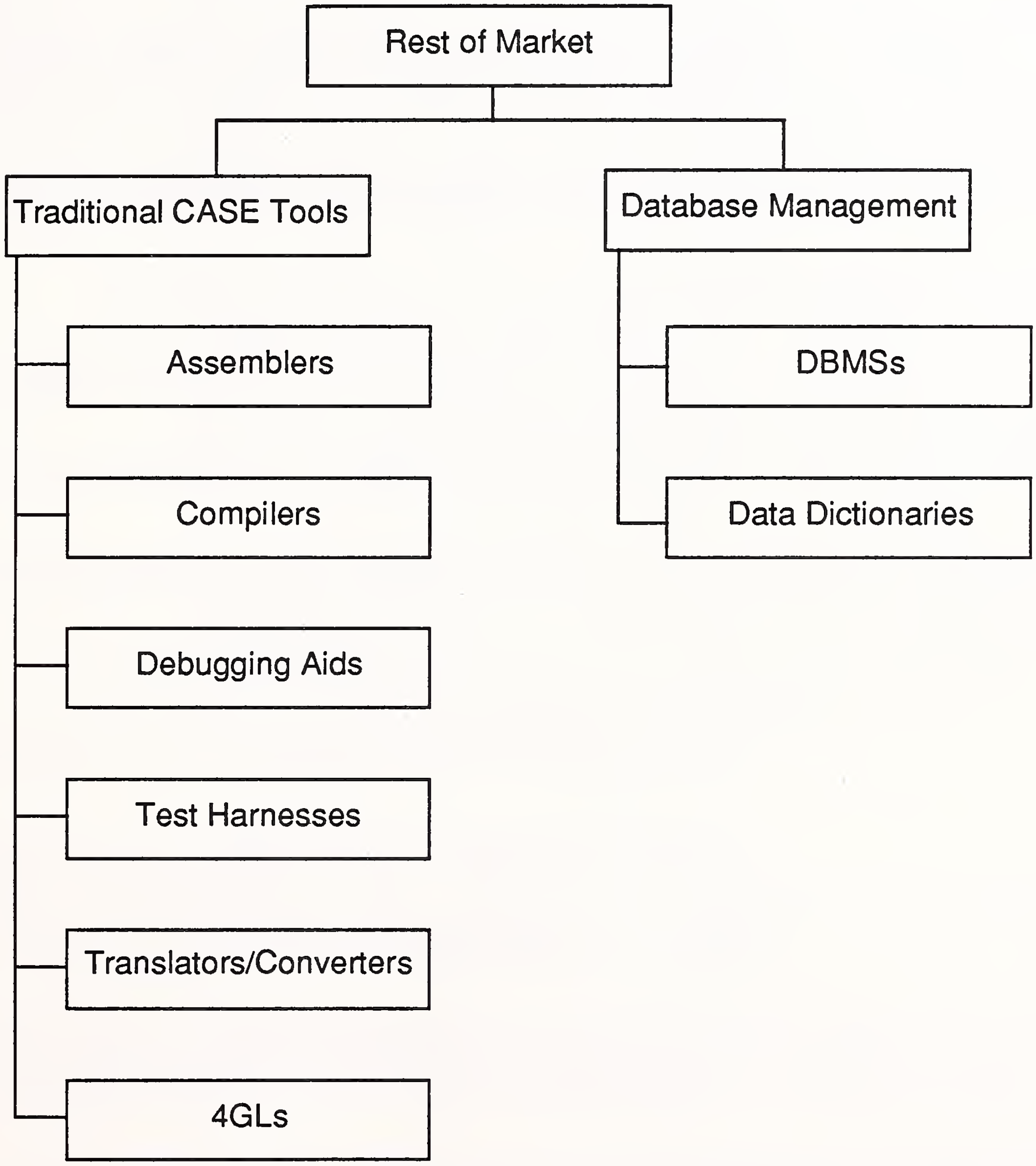


EXHIBIT III-3

STRUCTURE OF THE REST OF THE APPLICATION DEVELOPMENT TOOLS MARKET



The overall application development software market for both products and associated services (maintenance, education, training, consultancy and implementation services) in Western Europe for 1987 was $1,965 million, and INPUT forecasts that it will grow from this base to $7,900 million by 1993. By 1993, the products portion will be the biggest single sector in the $25,000 million software products market.

The current growth rate of this market was 28% (between 1987 and 1988), but key segments are growing much faster—the front-end tools segment, for example, grew as fast as 75% for the year.

Over the 5-year forward period, the overall growth rate of the sector will be 26%, with products growing at 24% per annum on average and services by as much as 36% per annum.

The principal driving forces fueling this growth are:

- The gap between the demand for software and the supply of skilled professionals to produce it
- The growing size and complexity of major software projects in the systems integration business
- The increasing reliance upon software in both commercial and industrial systems
- The safety-critical nature of major industrial systems
- The growing realisation that professionals must be given quality tools to finish the job

Alongside these market stimulants there are a number of factors slowing growth in the short-term. However, 1988 has seen a steady erosion of their importance, especially in certain country markets, such as the U.K., Belgium, the Netherlands and Scandinavia. Reasons are:

- Confusion as to the true direction of software engineering, because of the large number of products and methods all shouting deceptively similar but, in practical terms to the user, different stories
- Cultural resistance to radical changes in working practices in the IS department
- Perception of CASE systems as still high-cost and hard to justify
- Variations on the NIH (Not Invented Here) syndrome, usually stemming from perceptions of new approaches as "not sufficiently flexible to cover our requirements"

- The continuing tendancy of the industry to shoot itself in the foot by trying to do away with the software professional as an expensive, overindulged and unnecessary intermediary between the end user and the system the end user really wants.

B

The Market for Software Development Methodologies

Exhibit III-4 shows the INPUT market sizing and forecasts for the total market for software development aids in Europe. The chart includes products in the top half of the table, and services in the bottom half. The market for methodologies is associated principally with the services segment. Since most methodologies are public-domain information and can be obtained in published books, this sector consists of a mix of product and service to further the implementation of methods and CASE tools.

The market should be addressed as a service, with the main revenues coming from training and implementation.

INPUT forecasts that the combined sectors for these two will grow from $65 million in 1987 to $780 million by 1993. In other words, in five years' time this service sector alone will account for over $3/4 billion, without any product component.

The cause of this phenomenal growth is the difficulty of migrating from the present manual or non-CASE way of working for the many users who find themselves in the new-development-vs.-maintenance trap. This growth will present a multitude of service opportunities, ranging from:

- CASE Strategy Planning, through

- Tool and Methodology Selection, to

- Complete deals involving Implementation of CASE environments on a standard turnkey basis, or

- Operational management (or facilities management) deals whereby the service house takes over the old environment to allow the in-house teams to concentrate on introducing the new methods and tools

The keynote must be to stress the ability to tailor the methodology and the tools and environment needed to support it to the individual client company situation.

EXHIBIT III-4

APPLICATION DEVELOPMENT SOFTWARE PRODUCTS/SERVICES MARKET, WESTERN EUROPE 1988-1993

Sector	Market Forecast ($ Million)					
	1987	1988	AAGR '88/'87 (Percent)	1990	AAGR '93/88 (Percent)	1993
I-CASE & Integration Tools	50	70	+40	120	+30	260
Front-End (Upper CASE)	30	53	77	147	56	495
Back-End (Lower CASE) Comprising:	1620	2038	26	3160	22	5525
a. Code & Application Generators	50	85	70	224	49	630
b. 4GLs & DBMSs	1020	1275	25	1991	23	3575
c. Other Languages & Utilities	550	678	24	945	14	1320
Subtotal—Products	1700	2161	27	3427	24	6280
Maintenance	200	254	27	410	27	840
Education/Trg. (Methods/Tools)	35	50	43	125	52	400
Implementation Svces., Conscy.	30	46	53	107	53	380
Subtotal—Services	265	350	32	642	36	1620
Total	1965	2511	28	4069	26	7900

C

The Market for Software Tools

Exhibit III-4 shows that the 1987 market in Western Europe for these software products measured $1,700 million, was growing at a rate of 27% in 1988 and will reach $6,000 million by 1993. In addition, a software maintenance and support component will grow from $200 million in 1987 to over $800 million by 1993, as the installed base of products increases.

This market sizing and the forecasts include the software portion of standard turnkey systems sold in the I-CASE segment.

Exhibit III-4 divides the market sector into three components named after the section of the System Life Cycle addressed:

- Front-end products include all tools and systems aimed at the requirements analysis and system design stages.

- Back-end products include systems addressing the program and system construction stages, as well as the maintenance and enhancement phases of the system life cycle.

- I-CASE products include products that provide for the complete life cycle, from conception to the end of a system's life, and in particular cover project control, documentation and version control management.

The interfaces between front-end and back-end are defined to be firstly at the program specification level (during system development) and secondly when systems come forward for reanalysis and rewrite at the end of their lives. Full I-CASE products should allow for this wraparound effect at the end of each life. INPUT has not encountered any marketed products that do this and therefore, until the sector matures a little more, will loosen its definition to include systems such as the current generation of IPSEs that cover a single system development life cycle in its entirety.

The forecasts include both new and traditional CASE products, as described earlier in Section A of this chapter. The majority of the traditional segment value lies with 4GLs and DBMSs, which are going to be integrated into the life cycle approach over the course of the next five years.

The major trends affecting the products segment are:

- Integration of new products and enhanced versions of older products into the life cycle

- Development of standard interfaces to allow tools to be integrated and to interconnect—for example, the initiatives from the EC, ANSI, and

ISO in the matter of the PCTE (Portable Common Tool Environment), CAIS (Computer-Assisted Integration Standard) and IRDS (International Resource Dictionary Standard) draft standards respectively

- Development of interfacing components to allow unintegrated or incompatible tools to work together
- Separation of the development (source) machine environment from the production (target) machine environment in commercial IS departments
- Use of powerful workstation networks (LANs) based on 386 and 68000 standard hardware platforms to power the source environment
- Development of system encyclopaedias and compatible data dictionaries as a product subsegment to ease CASE migration problems

Besides the above technical considerations, the most important industry constraint, the numbers and quality of the DP professionals, will be eased by the firm separation of source and target environments. This is a prerequisite for finding a solution to the software crisis because:

- The source environment needs to become the proper seat of the software engineer.
- The industry must finally give up its knee-jerk reaction of trying periodically to engineer the professional out of a job.
- Software engineering must become the proper task of the professional, whether that professional is called a programmer, a designer, a business analyst, a knowledge engineer or any other title.

End-user desk-top computing will contine to be undertaken more and more in a co-operative processing mode between user and professional, with prototyping reaching the level of a true simulation exercise.

Exhibit III-5 shows the INPUT market sizing and forecasts for the software engineering sector by major country markets in Western Europe:

- West Germany is the largest market and will retain this leadership over the forecast period.
- U.K. and the rest of Europe are growing fastest in the short term, followed by France.
- Scandinavia and the Benelux countries are well attuned to the need for CASE, and for their sizes are well penetrated with product.

EXHIBIT III-5

APPLICATION DEVELOPMENT SOFTWARE PRODUCTS/SERVICES MARKET, WESTERN EUROPE 1988-1993 (By Country Market)

Country	Market Forecast ($ Million)					
	1987	1988	AAGR '88/'87 (Percent)	1990	AAGR '93/88 (Percent)	1993
West Germany	491	605	+23	970	+24	1780
France	425	547	29	810	25	1660
U.K.	400	530	33	870	25	1620
Italy	197	250	27	415	28	860
Benelux Countries	158	200	27	325	27	650
Scandinavia	138	175	27	285	25	530
Rest of Europe	155	203	31	395	32	800
Total	1964	2510	28	4070	26	7900

- Italy will grow faster than the other major countries but is starting from a relatively small base.

The country analysis in Chapter VIII will discuss the different country business climates and how they affect the appropriate product and delivery mode mix in each market.

Exhibit III-6 contains the exchange rate and inflation rates used in deriving the forecast tables in the two previous exhibits. The values in those tables have been expressed in constant 1988 U.S. dollars, converted at the rates shown.

EXHIBIT III-6

U.S. DOLLAR CONVERSION RATES AND INFLATION RATES BY COUNTRY IN WESTERN EUROPE

COUNTRY	CURRENCY	EXCHANGE RATE	INFLATION (Percent/Yr)
Austria	A	12.81	2.0
Belgium	BF	38.10	+1.0
Denmark	DK	6.90	-5.0
Finland	FM	4.34	-5.5
France	FF	6.13	+2.6
Ireland	IR£	0.68	-2.5
Italy	Lira	1351.00	+4.9
Netherlands	Dfl	2.05	+0.7
Norway	NK	6.66	-7.5
United Kingdom	£	0.59	+4.6
Spain	PT	121.40	+4.3
Sweden	SK	6.29	+6.9
Switzerland	S	1.51	+2.1
West Germany	DM	1.82	+1.0

*Midyear Rates for 1988

D

Indirect Market Opportunities

The main revenues pulled through by the emerging and maturing CASE sector will be in the hardware sector. Caused by the requirement for a separate development environment, this demand for hardware will fuel sales of:

- OS2-based PS/2 workstations and their compatibles
- UNIX-based 32-bit workstations such as IBM 6150, NCR Tower, Unisys, Sun and Apollo units
- Local-area networks offering file and encyclopaedia server capabilities, and their associated software

A range of potential opportunities will open for CASE capability offered by delivery modes other than the software-product-with-service mix:

- Turnkey systems will provide systems houses and VARs with the chance to offer tailored or packaged CASE to suit the individual company.

- Systems integration (SI) contracts can be offered that transfer users from annual coding to CASE technology or, as that technology matures, from one generation to another.

- Operations Management (FM) deals, in which a computing service vendor takes over a user's development environment while the user systems department migrates to a new CASE technology, will offer a set of make-or-buy options similar to those provided by the SI route.

- Processing services vendors will be able to offer CASE technology workshop facilities, in which a range of CASE methods and tools can be obtained for short- or long-term periods on trial-and-evaluation terms.

IV

Current Vendors of Methodologies

IV

Current Vendors of Methodologies

A The Sector Overview

Formal methodologies for system development grew up during the 1970s. These came as an extension of traditional good project practice for the implementation of software systems.

Prior to the evolution of system development methodologies, the best practice in the field depended upon standard methods and good project management. Typically, these methods consisted of breaking projects into a number of distinct phases, during which a series of tasks was performed that resulted in either written specifications or, at the final implementation stage, in the writing of program code. These stages were assisted by a number of techniques, many of which were diagramatic—e.g., systems flow charts, program flow charts and truth tables. This good practice was assisted by project control techniques derived from project management methods used on large capital projects.

During the 1960s the idea of building modular programmes had become prevalent. Modularity assisted with the task of controlling the complexity of systems. Setting a maximum module size for program code helped to deal with the overall size problem and the need to parcel the work among members of a software development team. This development coincided with the introduction of high-level languages that assisted with the building of self-contained subroutines and procedures.

All these methods were supported by standards, which dealt with:

- The man-machine interface
- File formats
- Consistency of project documentation
- Etc.

The need for more formal methodologies came when the complexity and size of systems had reached a stage where some form of modelling of the behaviour of the system was necessary prior to design. This need went with the requirement of preventing slippage in terms of time and cost, and being able to test the system at the logical level before committing to construction.

This stage also coincided with a key technological development, i.e., the uptake of database technology that has lead to the evolution of a data-driven view of systems. At the same time formal structuring of program designs had been called for by acedemic researchers who were horrified at the proliferation of inelegant and therefore unmaintainable programs written without recourse to any formal techniques.

In summary, we can say that the flexibility given to system builders by the advent of software-driven intelligence had lead inevitably to an increase in situations that could be described as growing "like topsy". We see that by the beginning of the 1970s, both at the programming and the system design levels, there were moves to curb this urban sprawl by the rigid adoption of formal (one would hope, eventually mathematically based) techniques.

Specialist consultancies in methodologies have entered the market with their proprietary products and ancillary services. These methodologies have been encapsulated in books, articles and courses, and have been sold in these forms packaged with both consultancy and other implementation services. In this chapter our report profiles seven consulting or systems companies that have entered this market-place. Five of them are European in origin, indicating the higher interest in formality attendant on the European approach to system problems. This is in contrast to the new world approach, which is intrinsically more pragmatic.

These companies started to promote methodologies for a number of reasons and in a number of ways:

- To fulfill an assignment for a particular client, i.e., to introduce a standard methodology in a certain industry sector, such as central government in the U.K. or the military defence systems market in the U.S.A.

- To gain further consultancy assignments to implement the methodology in other organisations

- To promote training and education activities

At first all the methodologies were promoted as paper-based systems. To date, most of these specialist companies' revenues have been from consulting, training, and implementation services, i.e., they have all been in the professional services sector of the market. They do not necessarily start from scratch and wipe out all the good work of the preceding methods and standards. Rather, they tend to take certain techniques and build round them with greater formality and greater completeness.

By the mid-1980s these specialist vendors were becoming drawn into contact with the CASE tools vendors who were promoting products to assist software developers by automating certain common systems development techniques. These tools were fragmented in their solutions to software development problems. Tools had arisen over a period of 10-15 years, during which time the overall approach to methods had become that much more rigorous and that much more complete. Now, therefore, we find the specialist consultancies adapting their marketing stances to include tools to support their own methodologies, and these product revenues have now become a separate and significant revenue stream in the portfolio of products and services.

It is estimated by INPUT that at least 50 methodologies are in current use in Europe. Some of these are proprietary systems; some are in use only in-house by systems houses or in user IS departments. At least one hundred techniques that form part or are used somewhere in the methodologies (and probably twice this number) could be discovered.

The Western European market for professional services associated with these methodologies was in 1987 over $100 million, split approximately 60:40 in favour of training and education versus services. This figure does not include the supporting products, which were sold by the specialist methods houses, since these products are sold on licence as part of the CASE tools segment of the software products sector.

B

Michael Jackson U.K.

1. Company Background

Michael Jackson UK forms the largest unit in the Jackson Group of companies. This group has been named after the originator of the well-known structured programming method. Founded in 1971, the company is now a subsidiary of IDK Data of Goteborg, Sweden. The U.K branch is the main company in the group and has 32 of the staff of more than 70. The rest of the capability resides in Sweden, France and the Benelux countries, with distributors in certain other countries. In 1988 a West Coast branch was formed in the U.S.A, at Santa Cruz, California.

Michael Jackson takes as its mission in life (and its approach is somewhat missionary in nature) the promotion and dissemination of sound techniques for system design. These techniques are all based on a formal and rigorously applied methodology.

2. Systems Engineering Products

The methods adopted by Michael Jackson stem from the adoption of the structured program concept. This concept is derived from the ideal program structure in which the structure reflects the logic of the problem that the program is designed to solve. The concept therefore is that proper design will result in an elegant, concise and understandable program structure.

The methodology starts with the Jackson Structured Programme (J.S.P) method. This is taught in a series of courses covering the method for programmers, analysts and consultants.

The Jackson methodology goes beyond the programming stage and into system analysis and design. The mnemonic for this part of the methodology is JSD (The Jackson System Development) method.

The Jackson system methods are thought to be the most formal in the market-place. They start from the basic elements of program design and, if implemented fully and rigorously, can produce the final object code executed in the computer system. Working outwards from that starting point, the Jackson method is dealing with the definable building-bricks of the system. Some people call this the bottom-up approach, but Michael Jackson prefers to call it the middle-out approach.

The Jackson method insists on settling the design of a system or program before proceeding to its implementation. This sound principle is key to the success of this methodology, which has been very widely adopted in programming circles and may be incorporated as a set of techniques into other methodologies used in-house or marketed as loose umbrella concepts.

The method gives program designers their true worth. It makes program design an essential task. The program is the atom of a system, and program design and construction are the professional activities of the professional programmer.

There is an analogy with mechanical engineering, in which one first designs a part and then machines it. It is possible to assist the design of the part by computer automation. It is possible to automate the machining of the part; in fact, the machining may be done in a number of ways—manually (in an NC [numerical control] machine), automated or semiautomated. Again taking the parallel with mechanical engineering, the design must take into account the realities of the machining process, both its type and duration.

The Jackson Structured Programming method implies that programming should not disappear with the advent of new tools to assist in the machining or generation of program code. The accent should rather be placed on good design as a professional activity, and a debate can be conducted as to who is the best person in individual circumstances to do the construction—the designer, a separate programmer or a 'factory'.

Jackson Systems Design puts it all into the wider context of the system in which the program is a module. The company now has three revenues streams:

- Training
- CASE tools to help with implementing the method being taught
- Consultancy to assist in the adoption of these methods in-house

Exhibit IV-1 shows the structure of the course list that Michael Jackson runs. The systems design courses are the most numerous, the programming courses are the oldest and the courses on the tools teach effective use of both JSP and JSD.

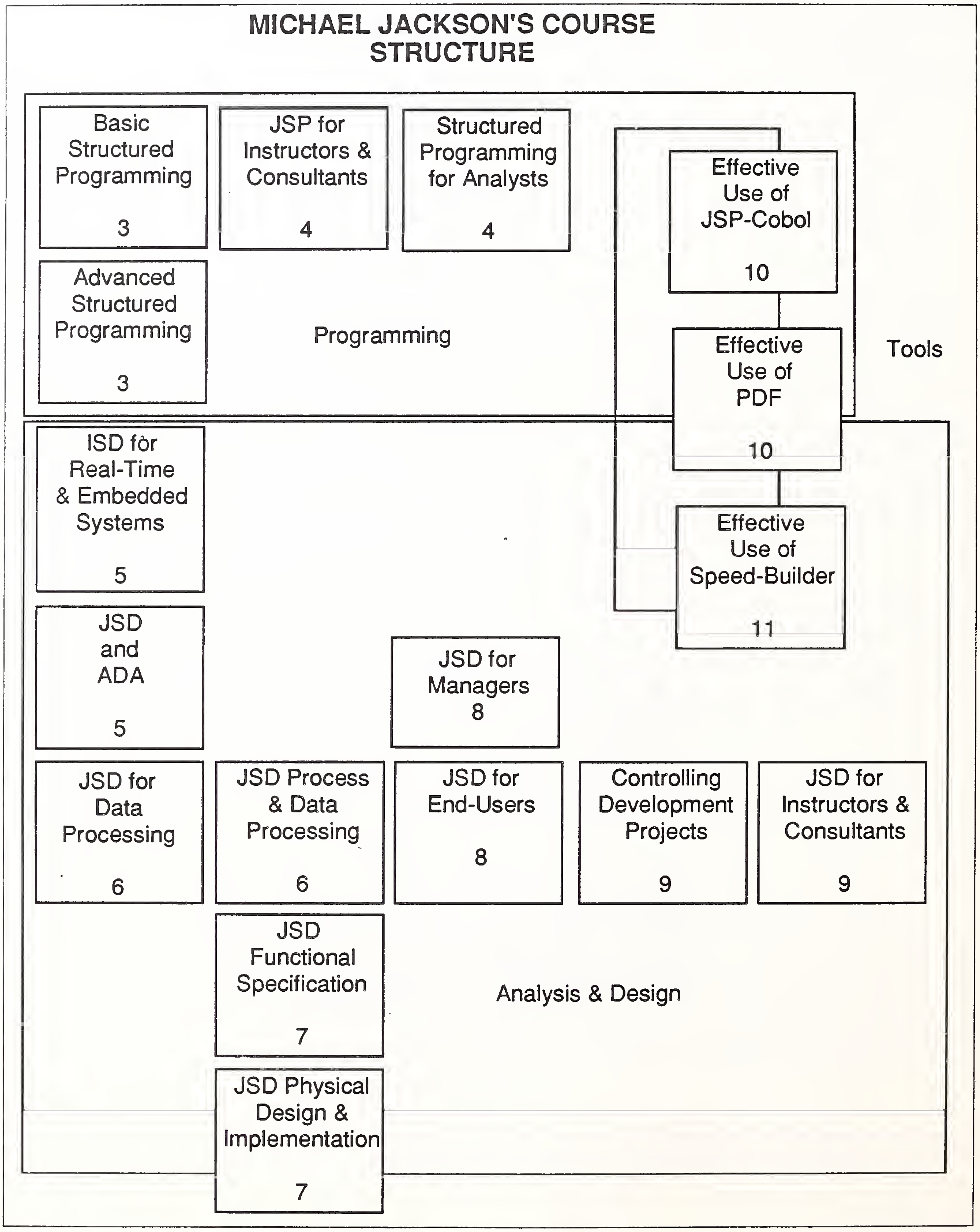
The group has now developed a set of CASE tools to support the implementation of methodologies. There are four tools:

- SPEED BUILDER is a tool for implementation of JSD. It follows the formal approach of the method by leading the user through the stages of analysis and design. Exhibit IV-2 illustrates the six stages of software development in which the Jackson method can be automated by use of these tools.

- JSP-TOOL is designed to support the JSP by automating programming from the program design stage on. It is based on a workstation and separates the programming function into the two distinct phases of Design and Code. Structure diagrams from the Jackson methodology are drawn and stored on a PC. Once the design has been approved, it is an automatic step to produce source code in almost any language for almost any target machine. JSP-TOOL can assist in the maintenance of programs at unit level.

- PDF is a tool equivalent to JSP-Tool, but PDF is available on the PC, on DEC VAX under VMS or UNIX, and on IBM mainframes. While JSP-Tool uses the graphic methods available on the standard IBM PC and its compatibles, PDF makes available the same graphical functions but using standard IBM 3270 and DEC VT-100 terminals.

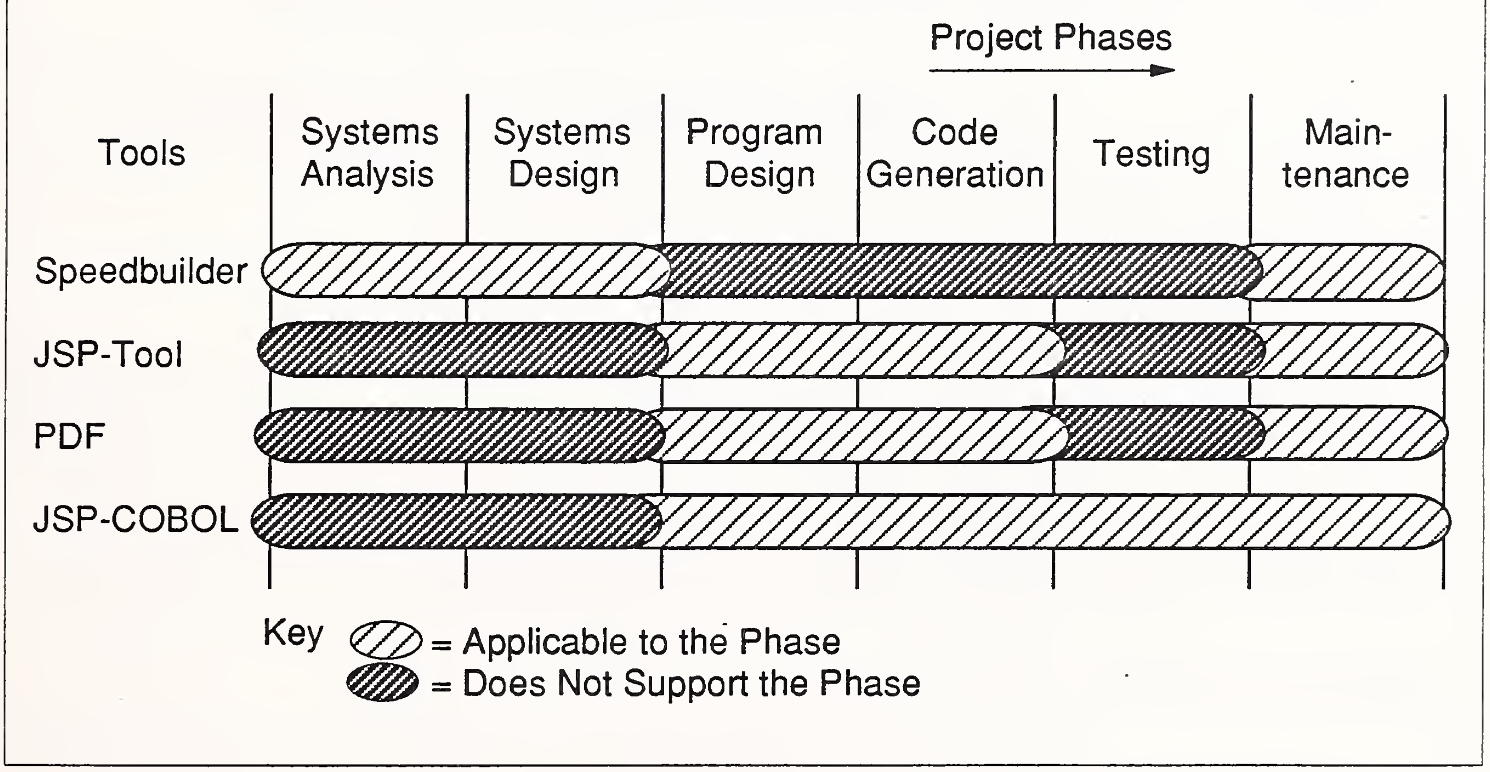
EXHIBIT IV-1


MICHAEL JACKSON'S COURSE STRUCTURE
Basic Structured Programming 3
JSP for Instructors & Consultants 4
Structured Programming for Analysts 4
Effective Use of JSP-Cobol 10
Advanced Structured Programming 3
Programming
Effective Use of PDF 10
Tools
ISD for Real-Time & Embedded Systems 5
Effective Use of Speed-Builder 11
JSD and ADA 5
JSD for Managers 8
JSD for Data Processing 6
JSD Process & Data Processing 6
JSD for End-Users 8
Controlling Development Projects 9
JSD for Instructors & Consultants 9
JSD Functional Specification 7
Analysis & Design
JSD Physical Design & Implementation 7

- JSP-COBOL, the final support tool in the Jackson quartet, is a COBOL preprocessor that automates the production of programs. It takes program designs in JSP Structured Text (or pseudo-code) as its input and produces programs ready for compilation, together with documentation. It is available on a wide variety of mini and mainframe computers as well as on PCs. As shown on Exhibit IV-2, JSP-COBOL has features that allow it to be used in the testing phase; this capability is not provided by the other tools described.

EXHIBIT IV-2

THE AUTOMATED JACKSON METHODOLOGY



3. Marketing

Michael Jackson UK operates a technical selling operation, which is mainly performed by the principals of the organisation, aided by the training consultants. Distributors assist in parts of the world where demand is not enough to warrant a separate subsidiary. Market coverage is assured throughout Europe, particularly in Scandinavia and the Benelux countries.

Growth of the organisation has recently been at the rate of more than 30% per annum. This speeding up of the company's activities is attributed to the cheapness of powerful hardware, on which the graphical support tools can now be mounted.

The company detects a constraint on its growth from the ability of the market place to absorb the full functionality of the methodology and its rigour. The method has been most favourably received in organisations looking for a rigorous approach to system development:

- The Defence industry sector.
- Although the methodology was adopted early in the commercial DP field, it is only recently that it has started to be reinforced there by the take-up of the support tools.

The marketing approach of the company has been characterised by strong brand differentiation. The company is now seeking to broaden its appeal across a wider commercial and financial market place.

This broadening will require careful management of the product/service mix. The company is, however, well positioned to capitalise on its balanced portfolio and its successful track record—if it can move into a phase of being more sales orientated.

C Yourdon International

1. Company Background

During the 1970s it became increasingly clear that the building of useful systems was not just a matter of writing bug-free programs. The need to produce programs that were easily understood and maintainable lead to the adoption of structured programming principles and the associated modular program design techniques. However, it was still found that systems were being built that did not meet the users' real needs, or the users' needs had significantly changed by the time the systems were completed.

Therefore during the early 1970s a new discipline, requirements analysis, arose. The aim of this discipline was to establish the true needs of the user before starting to build the system. The key development was the adoption of the "logical" model of the system. The logical model must provide a clear view of the system and be independent of the implementation technology.

In 1974 Ed Yourdon founded his company to develop techniques for the application of structured methods to requirements and systems analysis. The firm developed these techniques and now provides consultancy and training in their use.

Yourdon therefore was one of the pioneers of the development tools now known as structured analysis and design techniques. They are now widely recognised, accepted and used as a DP standard in both public and private sectors.

The company is currently running at a sales revenue level of approximately $10 million, of which one-third is in Europe. Outside the United States the activities are pursued under the name of Yourdon International Limited. The presence in Europe dates from 1983.

In 1984 other techniques developed by Ward and Mellor were incorporated into the Yourdon method to handle real-time systems.

In 1986 Yourdon introduced the Yourdon Analyst/Designer Tool Kit, a PC-based entrant into the CASE market.

In 1988 the organisation was acquired by the Eastman Kodak Corporation. It has continued to pursue an independent path within that group, although the purpose of its acquisition in strategic terms will perhaps become clearer with time.

2. Systems Engineering Products

Techniques developed by Yourdon are collectively known as the Yourdon Structured Method (YSM). These methods are public-domain information and are taught not only in Yourdon's own courses but by many institutes and colleges throughout the world.

The fundamentals of the Yourdon system are contained in the Yourdon model of system behaviour illustrated in Exhibit IV-3. This essentially Cartesian diagram is used to introduce the three dimensions of complexity envisaged in the model of system behaviour:

- On the X axis is the complexity of TIME-DEPENDENT behaviour, modelled using the state transition diagram (STD).

- On the Y axis is the complexity of DATA PROCESSING behaviour, modelled using the data flow diagram (DFD).

- On the Z axis is the complexity of the INFORMATION, this time modelled using the entity relationship diagram (ERD).

The relative proportion of complexity along the three axes represents, for any specific individual system, that system's footprint.

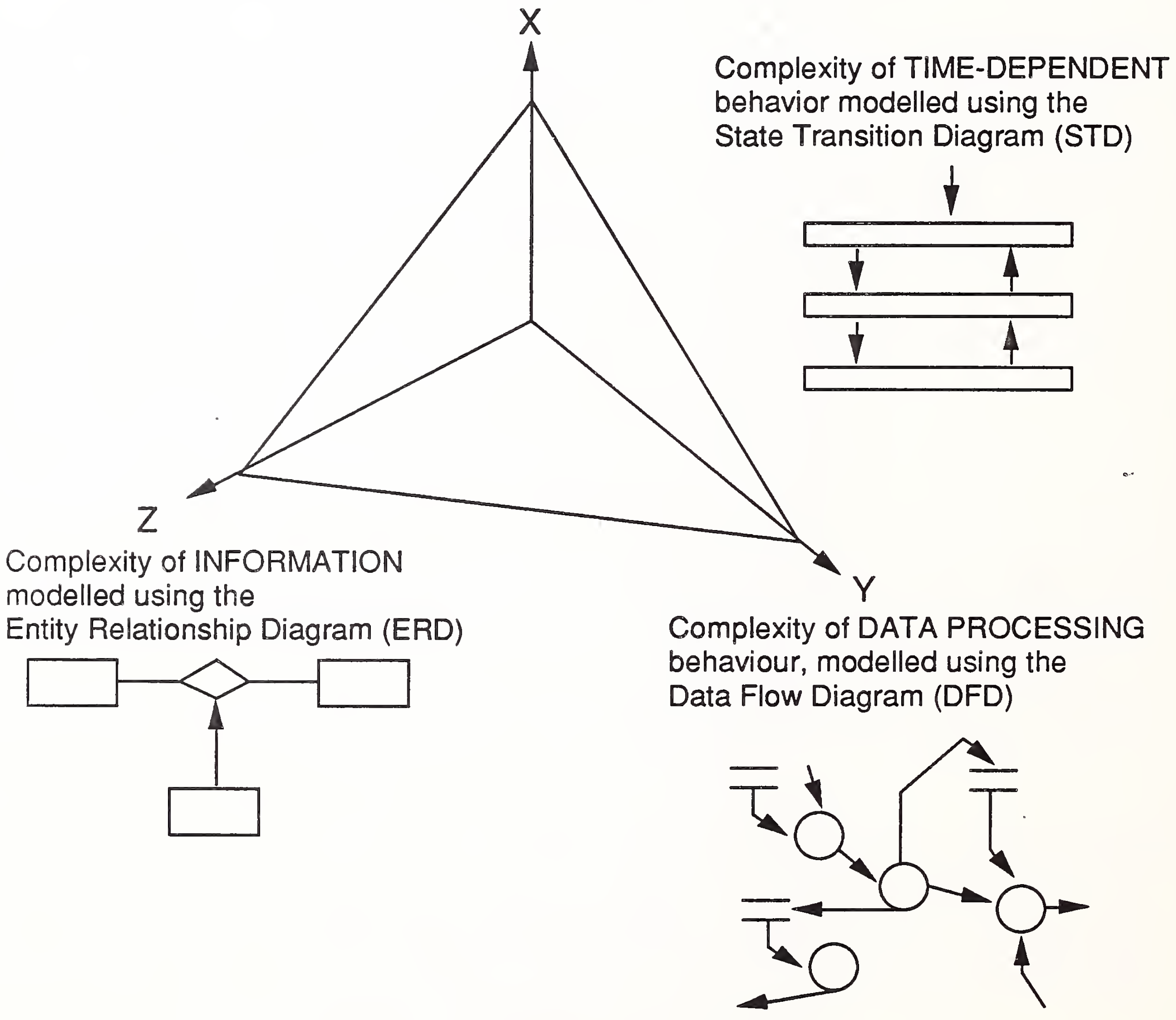
Yourdon is currently handling three revenues streams:

- Training
- Consultancy
- CASE Tool Kit Licences

The relative proportions in Europe of these streams are 60:30:10. This compares with the current ratio in the U.S.A of 40:30:30. Note the

EXHIBIT IV-3

THE YOURDON MODELS OF SYSTEM BEHAVIOR


X
Complexity of TIME-DEPENDENT behavior modelled using the State Transition Diagram (STD)
Z
Complexity of INFORMATION modelled using the Entity Relationship Diagram (ERD)
Y
Complexity of DATA PROCESSING behaviour, modelled using the Data Flow Diagram (DFD)

increased element of consultancy in the U.S. pattern; this increase is expected to occur soon in the European scene, probably within one to two years.

Consultancy amounts to assistance in the implementation of YSM or other general methodologies incorporating a range of techniques, or assistance with the implementation of the tool kit.

The Analyst/Designer Tool Kit is a PC-based system sold for just under $4,000 and available on PC-XT, AT and PS/2 systems or compatibles.

The basic functions of the tool kit are :

- To assist with the graphical drawing and editing of all types of diagrams used in the methodology
- To allow for verification of the correctness and consistency of these diagrams in relation to their accuracy in representing the system using the structured techniques
- To assist with the attachment of text to the diagrams and the composition of project documentation from the system as held in the machine's project dictionary

The tool kit has a feature that allows it to be modified to fit in-house or other proprietary methodologies.

3. Marketing

Yourdon has a twelve-member sales and marketing staff in Europe, supplemented by tele-sales and distributors, as well as the 'consulting sell' resulting from training courses and consultancy assignments.

It is anticipated that Yourdon will internationalise its product and consultancy capabilities to increase its penetration in other countries of Western Europe outside the U.K. In this context distributorship or joint agreements with hardware suppliers could be profitable.

Product development will include growth in consultancy. Increasing emphasis will be put on developing expertise in the data structure dimension of the Yourdon model. Currently Yourdon is perceived as very strong on the DFD and Time-Dependent Axes of the behaviour model, but less strong on the data structure axis, which is of increasing importance in the general commercial DP field.

Yourdon would be well-advised to stay in the front-end or "upper CASE" segment of the market for the present, developing its product capabilities and enhancing both professional selling and the product development programme. A particular area where the firm's profile would lend itself to enhanced sales is general consulting in the use of methodologies and even meta-methodologies for large groups that need to put an envelope round the use of multiple methods.

The company has a good future, but the company strategy is going to be complex to manage.

D

LBMS (Learmonth and Burchett Management Systems Plc)

1. Company Background

LBMS is a UK-based management consultancy founded in 1977 and now consisting of over two-hundred staff and claiming 1987/88 sales revenues of $17 million. The company was founded to specialise in the marketing of information systems development methods. It now earns around twenty percent of its sales in the U.S.A and is starting to move out of its bases in the U.K. and Eire into other world markets, including continental Europe.

The company has been growing between thirty and forty percent in revenue terms over the last few years. Its staff numbers have increased by approximately fifty percent in the U.K. over the last year. It is the market leader in the U.K. methods sector.

2. Systems Engineering Products

In common with those of other methodology vendors, the proprietary methodologies, of which there are four in the case of LBMS, are not sold as products. Going on a training course entitles the user to use the methodology taught, but LBMS actively markets the methodologies as products in the form of an implementation pack that consists of:

- Consultancy
- Training
- Automated support tools

LBMS regards the above as a practical approach geared to successful implementation. It is based upon a realisation that adoption of more exhaustive and rigorous formal methods is not going to happen overnight in large companies in which whole system development cultures may need to be changed.

The products currently being marketed as methods and support tools fall into a group under the umbrella term *information management*. There are four areas covered by the products :

- Systems planning
- Software engineering
- Project management
- Office systems

The methodologies marketed in each area are:

- LEAP
- LSDM (SSADM)

- PROMPT
- LOTAM

The corresponding software support tools are:

- Super-Mate
- Auto-Mate Plus
- PROMPT
- Office-Mate

The organisation is currently revamping the product catalogue and the methodologies are likely to change their names to:

- LBMS/Strategic Planning
- LBMS/Software Engineering
- Etc.

INPUT supports use of the term *information management*, because INPUT recognises that, in an organisational system environment, management is exceedingly important. Management implies that people are involved; good management normally hinges on good people management. The term has the advantage over its rival term, *information engineering*. Information engineering implies the analogy to a full engineering discipline, which in turn implies a controlled manufacturing environment and a fully definable product. In the real world of commerce, industry and government, in which actual systems must work, things are not as controlled as the analogy assumes.

The LEAP (LBMS Enterprise Analysis and Planning) methodology is aimed at providing a mechanism for determining how IT and IS can be used to shape and support the fundamental goals of an organisation. The end product is the production of a viable strategic plan. Exhibit IV-4 shows the structure of the information processed by the LEAP methodology.

The information systems methodology—LSDM—is also known by its brother name, SSADM. Originally the SSADM methodology was developed in conjunction with the U.K.'s central government computer-buying agency, CCTA (Central Computer and Telecommunications Agency). SSADM has now been adopted as the standard methodology for use in all U.K central government projects and is proprietary to this client. LSDM is the method proprietary to LBMS that is marketed commercially and undergoes constant product enhancement. It is aimed at the development of mainframe and minicomputer or other major systems projects of a reasonable size, and is based upon a six-stage process:

- Systems Analysis—the first three
- Systems Design—the second three

EXHIBIT IV-4

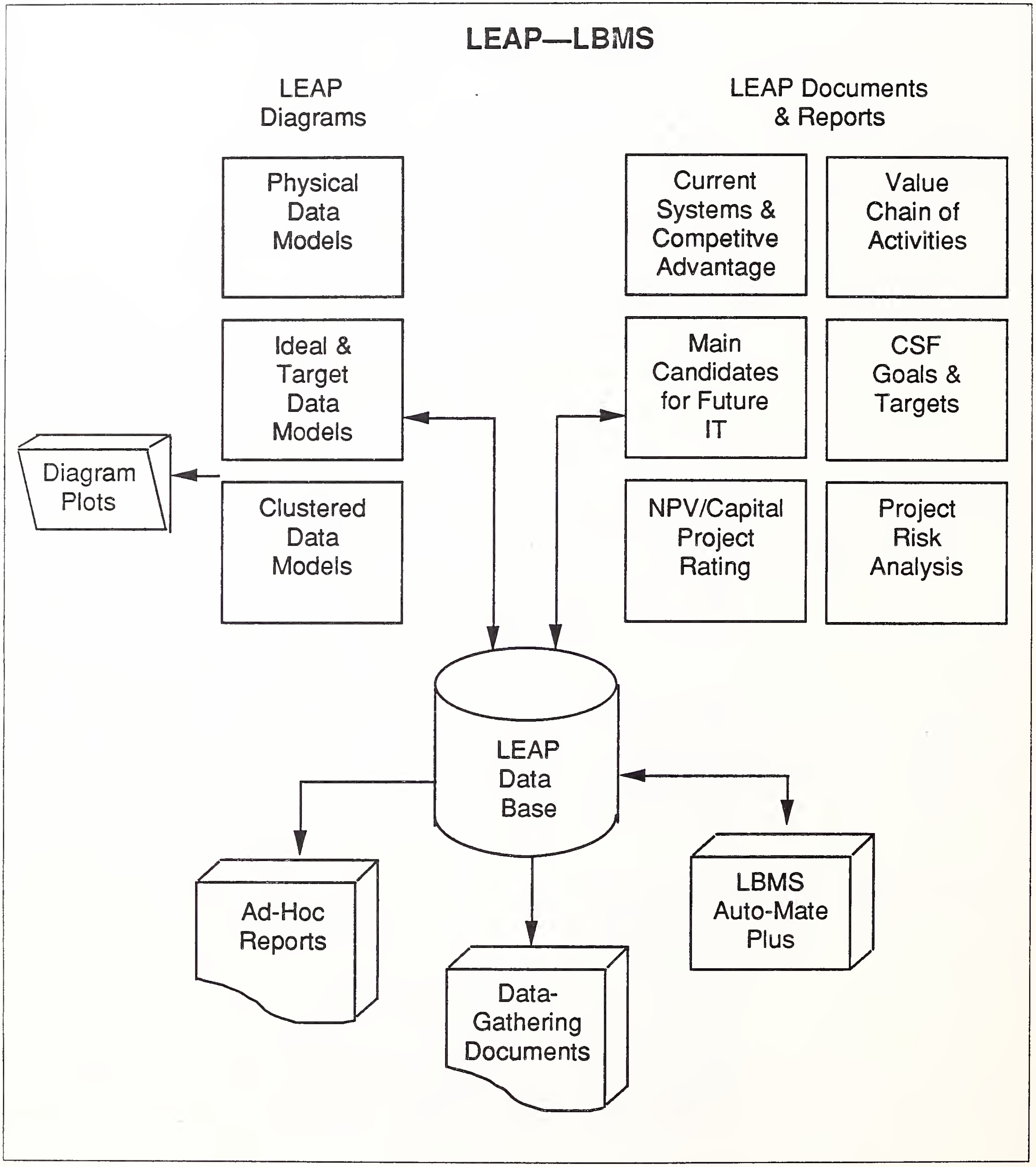
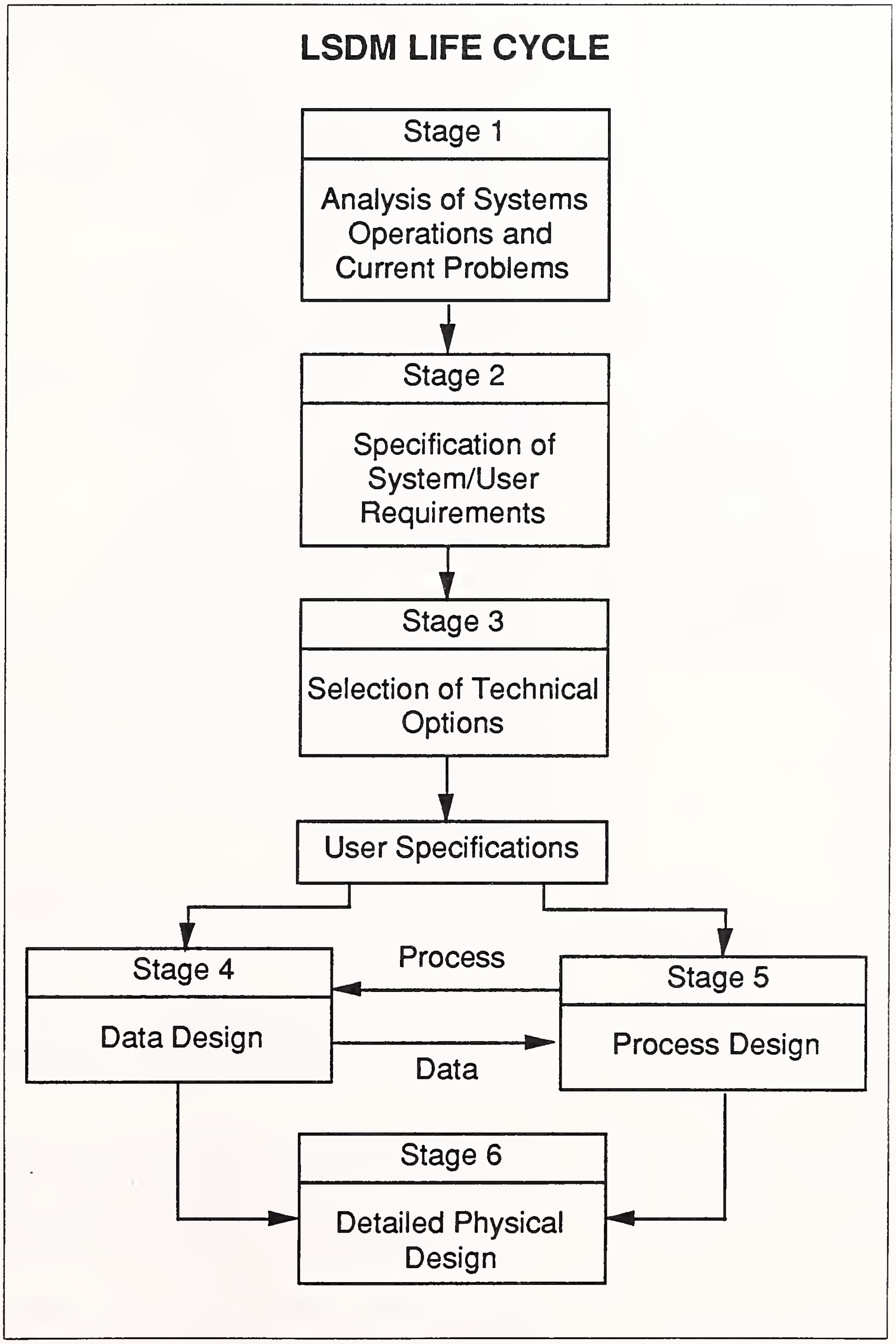

LEAP—LBMS
LEAP Diagrams
LEAP Documents & Reports
Physical Data Models
Ideal & Target Data Models
Clustered Data Models
Diagram Plots
Current Systems & Competitve Advantage
Value Chain of Activities
Main Candidates for Future IT
CSF Goals & Targets
NPV/Capital Project Rating
Project Risk Analysis
LEAP Data Base
Ad-Hoc Reports
Data-Gathering Documents
LBMS Auto-Mate Plus

Exhibit IV-5 shows the flow of the methodology from stage one to stage six, with the user specifications as an intermediate interface between the first three stages and the second three.

EXHIBIT IV-5



The LSDM method consists of a number of techniques, rules and guidelines for users of the techniques, standard forms for documentation and an overall framework defining the interfaces, the review points and the contents of working documents.

Eight important techniques are used in the method:

- Logical data structuring (LDST)
- Data flow diagrams (DFD)
- Entity life histories (ELH)
- Process outlines
- Relational data analysis
- First-cut data design
- First-cut program
- Physical design control (PDC)

Like the Yourdon method, LSDM gives three views of any system being modelled:

- The data structure view
- The data flow view
- The processing cycle view

Unlike Yourdon, the LSDM method puts the emphasis on the derivation of generic underlying data structures. LSDM puts emphasis here because in an organisation data structures tend to change little over time, although the processing requirements may vary considerably in terms of what is required and how it is to be accomplished. For this reason, the advocates of the database view claim that it is more important to establish the correct data structures as the primary view in commercial DP systems development.

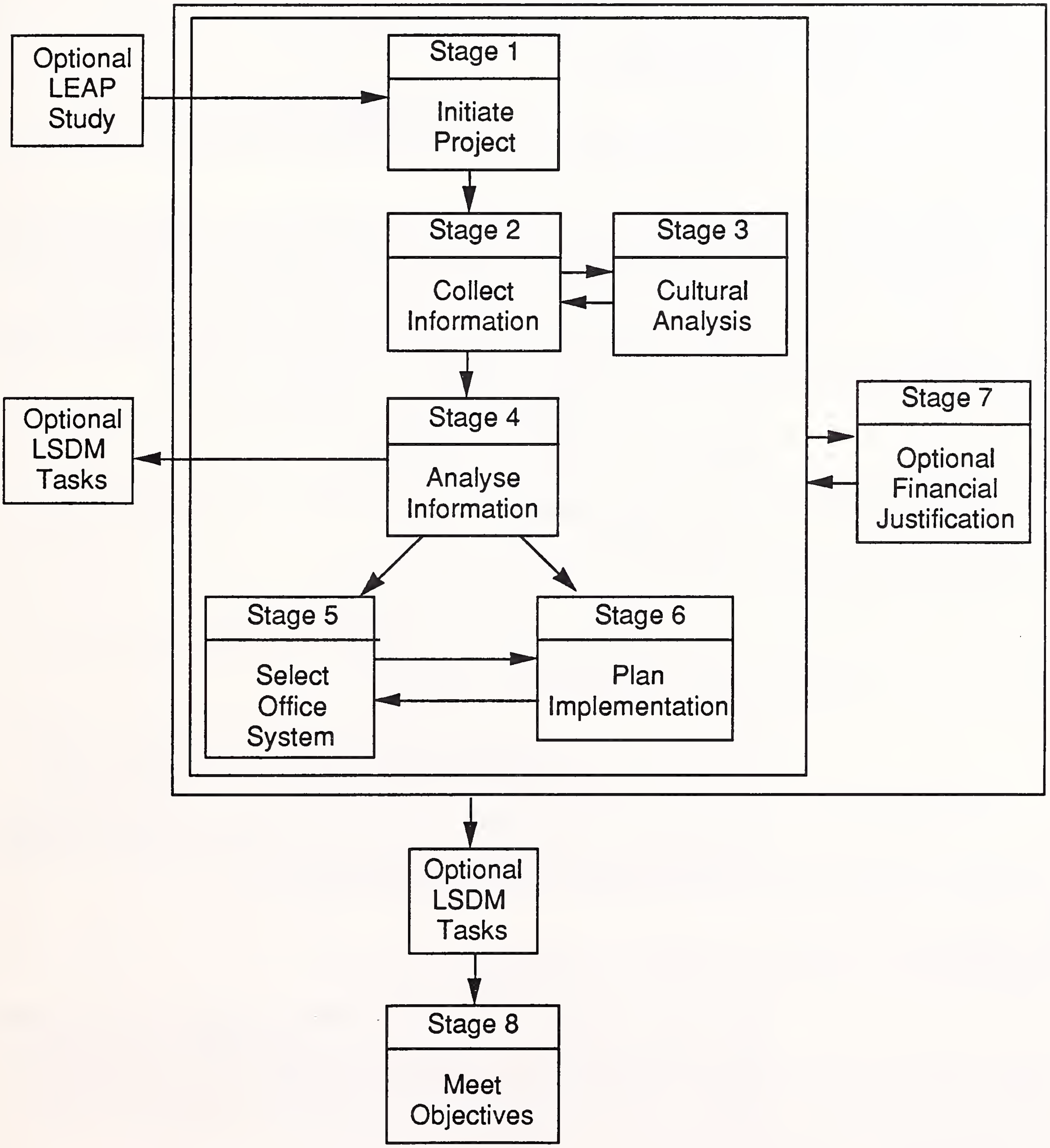
INPUT notes here an example of the revolutionary nature of information systems work. The slower variability with time of data structures can be partly attributed to the difficulty of changing them given previously nonautomated methods. The adoption of automated formal methodologies based on data-structured views of commercial businesses will encourage users to change their data structures more frequently because the new systems offer them previously unavailable flexibility.

This is a case of the system providing the facility to question the assumption upon which its own efficiency rests. This principal of "feedback to the problem areas," which is generated by automated solutions, is central to understanding the addictive effect of adopting automation.

LOTAM (LBMS Office Technology Analysis Method) is the most recent of the product offerings added to the LBMS catalogue. Exhibit IV-6 shows a summary of the entire LOTAM method.

EXHIBIT IV-6

OUTLINE STRUCTURE OF LOTAM



In many ways the method is similar to the more traditional hardware and software selection processes undertaken by consultants or IS groups at feasibility or design study stages. The LOTAM method puts emphasis on the people dimension in effective office systems designs. It is still early in the implementation of the LOTAM techniques, and product support must and will evolve.

3. Marketing

The products are normally marketed as standard packages involving consulting, training, the software products themselves, and support and implementation services.

The way the services are offered implies a loosely coupled approach to the CASE market place. This is sometimes known as the Open Architecture Approach in which individual modules tackle individual tasks, interfaces between tools and methodologies, and tools. Tools are provided by different interfacing mechanisms, e.g., other specialist software tools or indeed manual methods, including office support.

The open framework approach allows for alliances to be formed in product development and in marketing:

- For example, a bridge between LSDM and Hoskyns' PMW Project Control System

- Support for the joint development initiative of IBM

- A licensing agreement with Cullinet for the marketing of the Auto-Mate Plus product in the U.S.A.

In summary, we can see that LBMS is going forward with very useful growth rates and a balanced mix of product and service between the CASE tool, the training in the methodology and the implementation services needed. The trick for success in the future must be to emphasise the component of the mix that is most in demand at any particular time, according to the changes in market requirements.

E
Sema Group

1. Company Background

The Sema Group is an Anglo-French company recently formed from the merger between Sema Metra of France and the CAP Group Plc, one of the U.K.'s major professional services companies. The combined revenues for 1987 were over $400 million and together they employ about 6,500 people.

Although Sema Metra has been growing at 19% per annum to double its revenues in the last four years, it is not expanding as fast as its long-standing opponent, Cap Gemini Sogeti (CGS). Sema Metra is number two in France to CGS, whose latest reported revenues exceeded $700 million, up over 40% for the previous year.

Sema Metra has historically had two almost-equal major revenues streams in—commercial data processing and scientific/technical systems—as well as two small separate streams in market studies and management consultancy.

2. Systems Engineering Products

Sema Metra is well known as the designer of the MERISE methodology (Methode Realisation de Systemes). The system was devised by Sema in conjunction with the French administration and other systems houses and consultancies.

Merise is public-domain information; it is taught in schools and used by project teams throughout France. Sema Metra now also markets and supports two other methodologies:

- Racines: this methodology is for Strategic Planning within large organisations (in France this activity is know as Schema Directeur)

- Axial: this is an alternative methodology adopted by IBM France to include more physical design content and to make up for the lack of this component in Merise

Merise views the information system of any organisation as a three-level structure. This structure is illustrated in Exhibit IV-7.

The three levels used are:

- At the top, the guidance system controls the whole organisation.

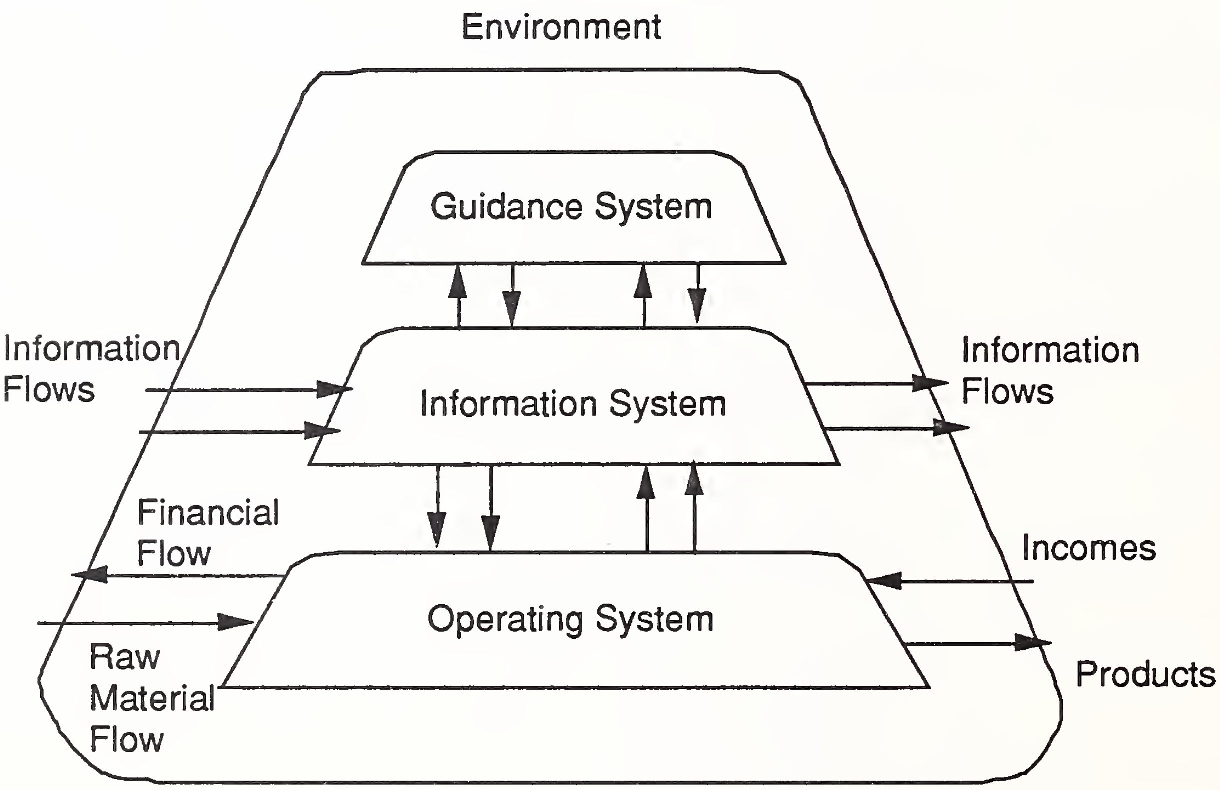
- In the centre level, the information system itself receives information from the outside world and shuttles it between the other two levels.

- At the bottom level in the structure is the operating system of the organisation, which is conceived as the part that carries out the functions for which the organisation was set up.

The Merise generalised conceptual picture is published with the blessing of the International Standards Organisation (ISO) and results from almost a decade of work initiated by studies on databases following publication of an ANSI report in 1975. The system attempts to put a general view on all information systems in any type of organisation, and this is in line with the French regard for conceptual order.

EXHIBIT IV-7

THE MERISE METHODOLOGY GLOBAL DESCRIPTION OF THE UNIVERSE OF DISCOURSE



Like many methodologies encountered in the INPUT study, Merise deals in trios of concepts. For example, the Merise system is conceived in a framework of three cycles:

- The abstraction cycle uses three database levels.
- The approval cycle recognises the need to identify the decision points during development of information systems.
- The life cycle, in common with any engineering project, consists of planning, study, implementation and maintenance.

Merise claims to be different from other methodologies in the accent it puts upon the abstraction cycle, which is itself conceived at three levels:

- The conceptual level is the result of identification of objectives and management decisions. It describes the classes of things and the rules of behaviour of the objects to which the designer must address himself, according to the objectives of the guidance system.

- The organisational level describes the nature of resources to be used for supporting the data systems required. These resources can be human or machine or a mixture of the two.

- The operational level results from technical decisions made according to the technical targets and constraints placed upon the information system.

The abstraction cycle is shown in schematic form in Exhibit IV-8.

The levels within the Merise methodology have their analogues in James Martin's Information Engineering Methodology (IEM), to be described in a later profile. However, the Merise methodology only claims to be an approach to the information systems to be developed. It does not contain a great deal of technical content at the lower levels. Perhaps for this reason, it has not been used much outside the French-speaking world.

3. Marketing

Sema Metra does not market the methodology as a specific service. Its association with the birth of methodology, however, is used as a sales tool to enable the company to secure assignments of all types.

Sema Metra markets and uses internally a number of tools to assist in the implementation of methodologies:

- Diamant III, a data dictionary for use with Merise

- Metradoc, a graphical and textual documentation aid

- MPM, a project management and control tool

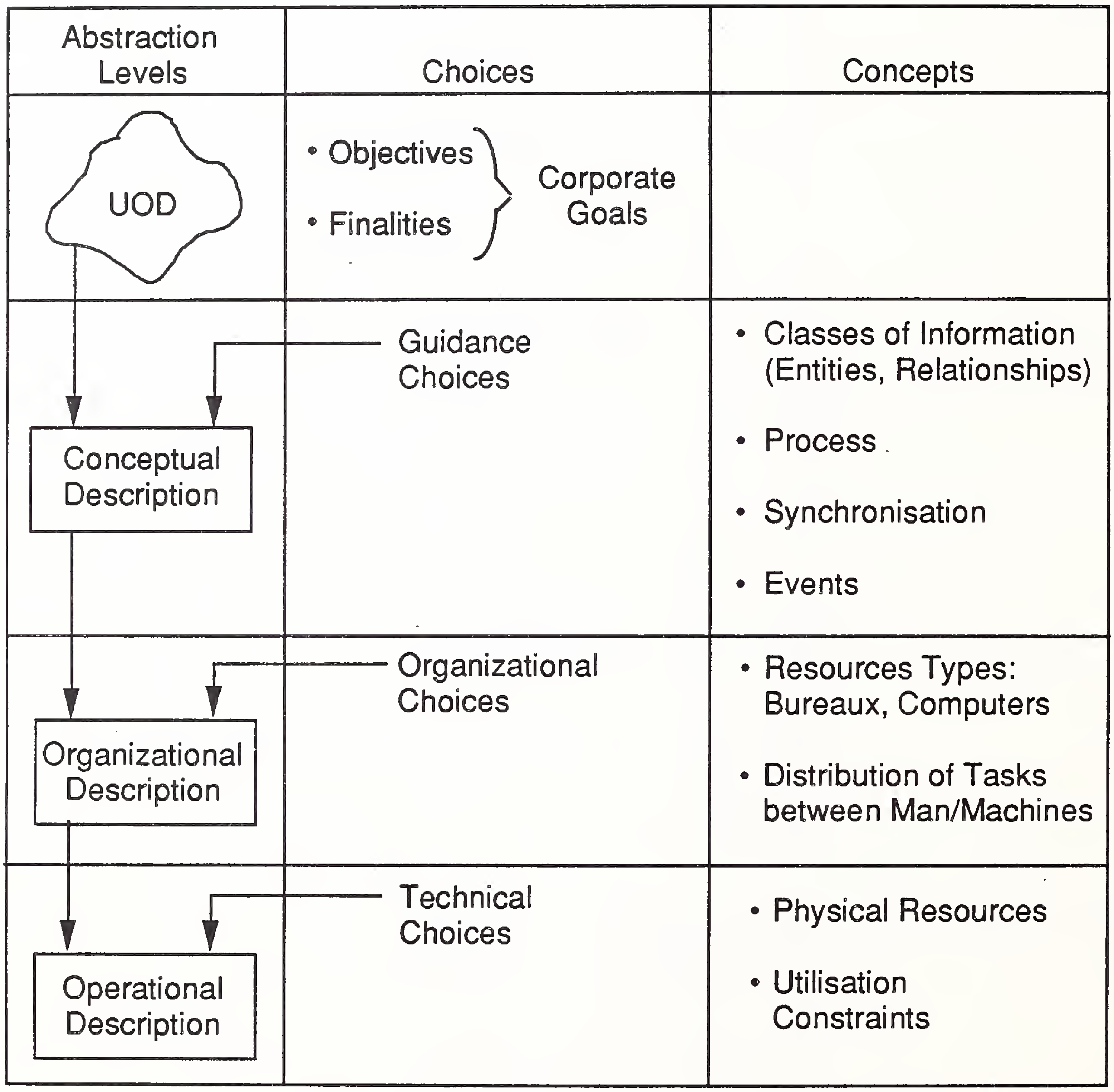
- USE.IT, a tool for specifying and automatically generating structured code

Sema Metra has not yet rationalised its activities in the field of methodology and CASE tools. Along with most French computer service companies, it does not recognise the market segmentations from the U.S.A. as strictly applicable within its own domestic sector.

With the recent merger the company should now realise that the time is right to grasp opportunities in the CASE field that capitalise on its past experience and its association with the Merise methodology. It will

EXHIBIT IV-8

MERISE METHODOLOGY
GROSS DESCRIPTION OF THE ABSTRACTION CYCLE



Abstraction Levels	Choices	Concepts
UOD	• Objectives • Finalities Corporate Goals	
Conceptual Description	Guidance Choices	• Classes of Information (Entities, Relationships) • Process • Synchronisation • Events
Organizational Description	Organizational Choices	• Resources Types: Bureaux, Computers • Distribution of Tasks between Man/Machines
Operational Description	Technical Choices	• Physical Resources • Utilisation Constraints

undoubtedly go through a period of settling down in which it will have to come to grips with the different characteristics of the two major countries in which it is based. The group will have to devise an overall product strategy that gains from the increased resources now available.

F

Arthur Andersen

1. Company Background

Although a relative newcomer to the CASE tools market, Arthur Andersen, as one of the world's largest accounting practices, has been in the computer systems development field for over twenty years. The firm can therefore claim to have been in the data processing business as long as anyone, both as a user and as a developer of systems for its clients.

Arthur Andersen is one of the big eight accounting firms. In its most recently published figures, worldwide revenues exceeded $2.5 billion. It ranks as number five in the European league of accounting firms.

The firm operates as three separate practices for audit, tax planning and management consultancy. The computer systems activities reside within the management consultancy practice and its divisions in each country operation. Worldwide management consultancy accounts for around $1 billion of total revenues.

Arthur Andersen has been through several phases of development with its computer systems involvement:

- The firm has always been in bespoke or tailored systems.

- When packages started to arrive in the mid to late 1960s, Arthur Andersen started to move into that field with systems in the accounting and, later, manufacturing fields.

- Its move into CASE tools has resulted from its experiences in developing systems (both bespoke and package-based) for its external clients, as well as for its own internal audit and tax planning functions. To give some idea of its involvement in computers, the firm reminds one that it is typically installing two major systems a day in the world.

There has been in the last three years a strong move on the part of the U.S. parent to enter the software products field and to capitalise on the development experience built up over the years by entering the new, popular CASE market. This entry has been done by setting up a division called Andersen Software.

2. Systems Engineering Products

The Arthur Andersen activities in this field are all contained within the Foundation product. Foundation is a combined set of methodologies and tools offering an integrated environment for software engineering. It claims to put equal emphasis on methodology and tools. This integrated environment approach is now being called I-CASE, standing for Integrated CASE. Arthur Andersen claims that Foundation is the only computer environment built specifically for IBM DB2 users and, moreover, that it is the only one that covers the entire life cycle from planning and design, through implementation, to systems and support.

Foundation consists of three components:

- Method/1
- Design/1
- Install/1

Method/1 is the master element in the scheme. This element is essentially a project management tool that fits flexibly into a range of methodologies and approaches. The firm's appproach to methodologies is to say that to get the best benefit from Foundation, clients should take all three modules. However, it is understood that methodologies will already be in place, and Method/1 is designed to be adaptable enough to fit into the client's framework.

Method/1 organises the development life cycle into phases. Within these phases there are segments, and within the segments there are tasks. Method/1 is based on the philosophy that routine check points are required to enable management to guide and review.

Method/1 recognises three different types of implementation:

- Iterative, which is required to make full use of AI techniques, 4GLs and prototyping

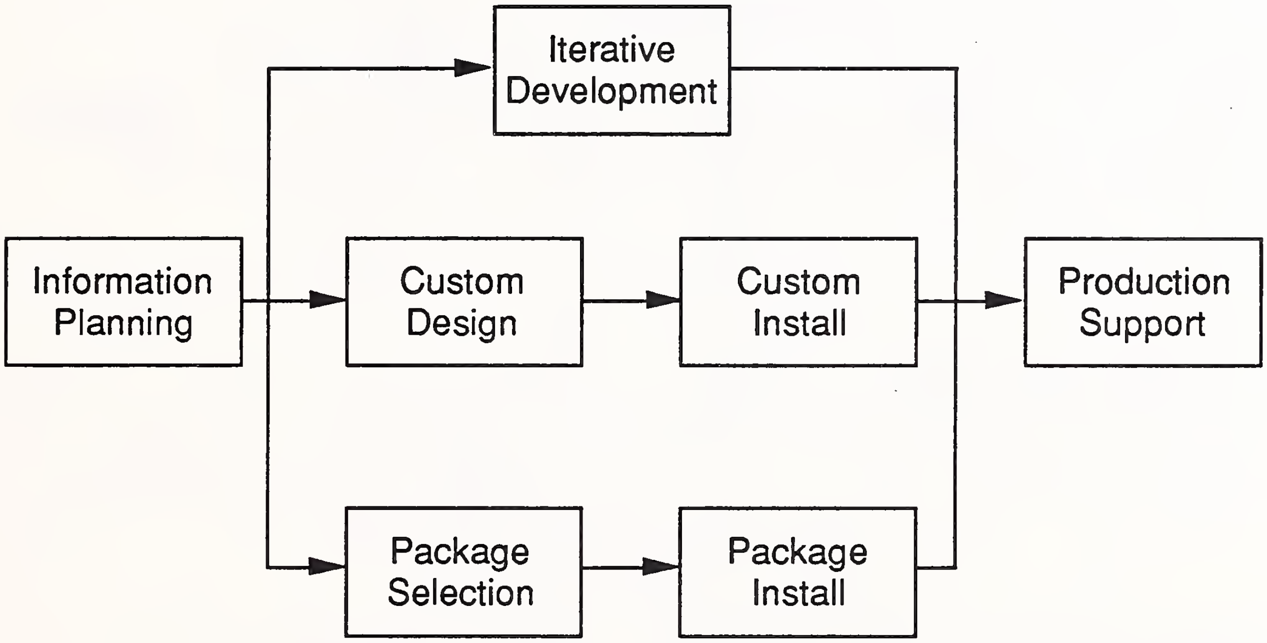
- Tailored software development

- Package selection and installation

The main types of development as supported by Method/1 are shown on Exhibit IV-9.

EXHIBIT IV-9

ARTHUR ANDERSEN'S METHOD/1— TYPES OF DEVELOPMENT SUPPORTED

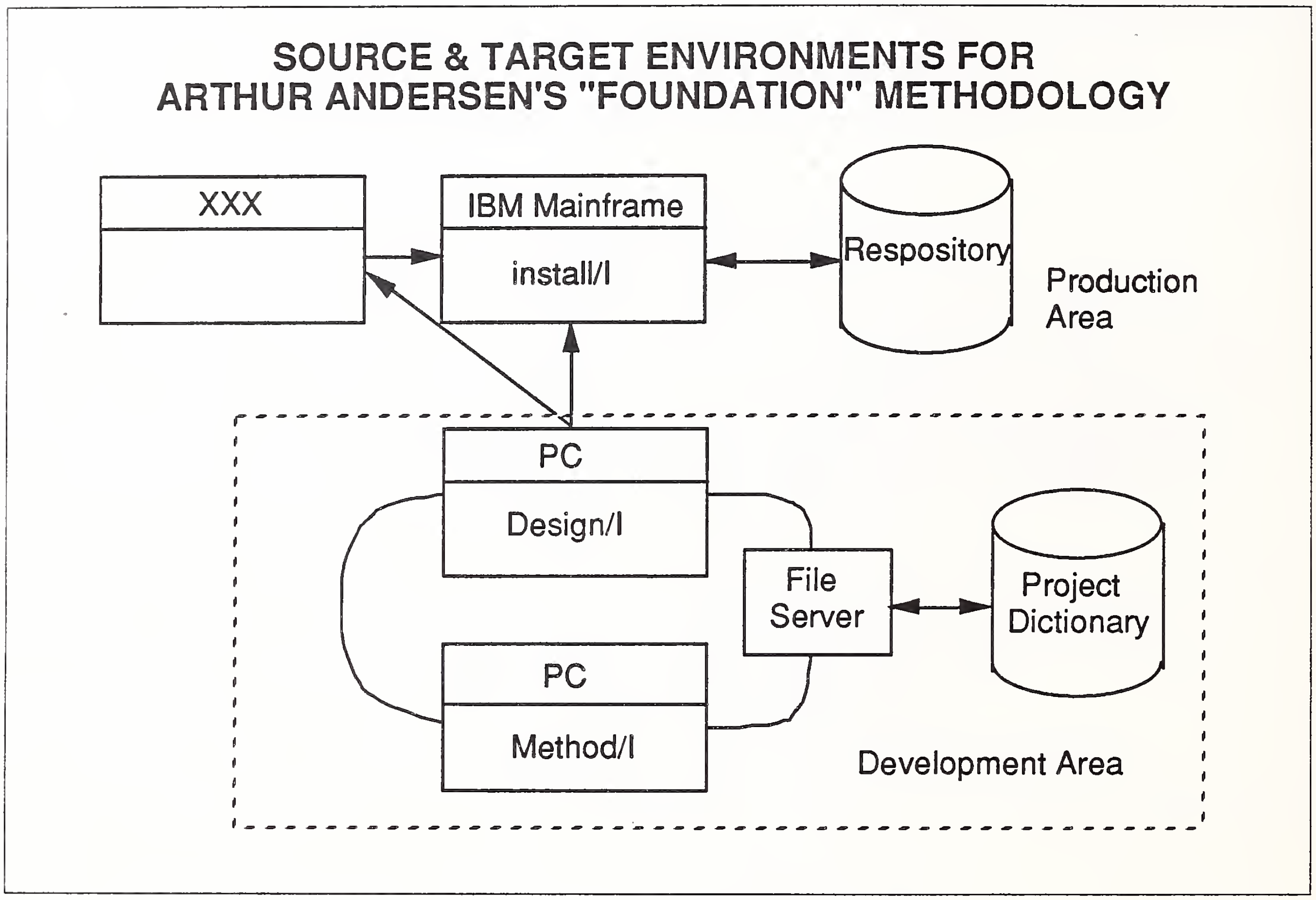


Design/1 and Install/1 cover the front-end and back-end, respectively, of the design and build cycle. Exhibit IV-10 illustrates the configuration on which the two support tool products operate. Design/1 is a PC-based product that can be used in standalone mode or on a local-area network. Sharing design data between users on a LAN encourages effective teamwork. As an application analysis and designer workbench, Design/1 is designed to be used in conjunction with the Install/1 module resident on the mainframe. Design/1 handles the interfaces to the DB2-based repository on the mainframe and can also interface to other data dictionaries—e.g., the Cullinet data dictionary.

Install/1 is a set of facilities to allow developers to implement and support high-volume transaction processing systems. Exhibit IV-10 illustrates the structure of the facilities, with the DB2-based repository in the centre of the architecture, allowing a series of functions appropriate to each phase of the life cycle to be supported.

As well as offering its extensive range of professional services and applications programming, Arthur Andersen also markets a customer

EXHIBIT IV-10



information system called CUSTOMER/1. This system has been built using the principles of the Foundation CASE environment.

3. Marketing

The Foundation system is an integrated CASE tool that retails at $150,000 and upwards.

It is therefore aimed at the larger companies and is considered to be in direct competition with James Martin's IEF (see next profile).

Arthur Andersen's marketing stance is to offer a complete range of system development tools, facilities and services—but product sales will need to be co-ordinated with the more traditional project sales. It is not yet clear how the market will react to such a broad range of offerings. The apparent advantage of Foundation over its main IEF rival is that it is based on a flexible approach allowing for integration of bottom-up and top-down approaches.

G James Martin Associates

1. Company Background

James Martin Associates was founded in 1980, and in 1981 started to market its Information Engineering Methodology (IEM). The concept goes back earlier than this, however, and the chairman of the company, James Martin, is responsible for coining the phrase *information engineering*. This phrase has become very important in the industry, as is the phrase *automation*, coined by another industry guru, John Diebold, in the early 1960s. The word and the concept both imply that information can be managed in the same way as the materials, components and systems produced by engineering disciplines. There is a mechanistic connotation to this word, which INPUT believes is inadequate to cover the full range of problems associated with the handling and management of information systems.

The move to implement the concept of information engineering as a full-blown methodology came in 1981, when the proprietary IEM was marketed initially as a paper-based system. The next phase of development came when James Martin Associates started to discuss with Texas Instruments (TI) the development of a complete tool set to implement the methodology in computer-based terms.

From 1983 to 1986 the CASE environment implementing the methodology was developed at the Texas Instruments development facility in the U.S.A. Thus was born the IEF (Information Engineering Facility). In 1985, TI delivered software based on a PC to selected customers for beta testing. Increased functionality was added to the product during 1986, and in 1987 the commercial availability of the IEF's Central Encyclopedia and Analysis Toolset was announced. In 1988 all of the IEF tool sets, including code and database generators, were made commercially available.

The company is now over 200 people strong, with offices in the U.S.A, the U.K. and continental Europe. Revenues for 1987 exceeded $18 million, an increase of over 70% on the previous year's sales. The U.K. contributed over 45% of this turnover, continental Europe over 40% and the U.S.A, which was the newest area of activity, about 13%.

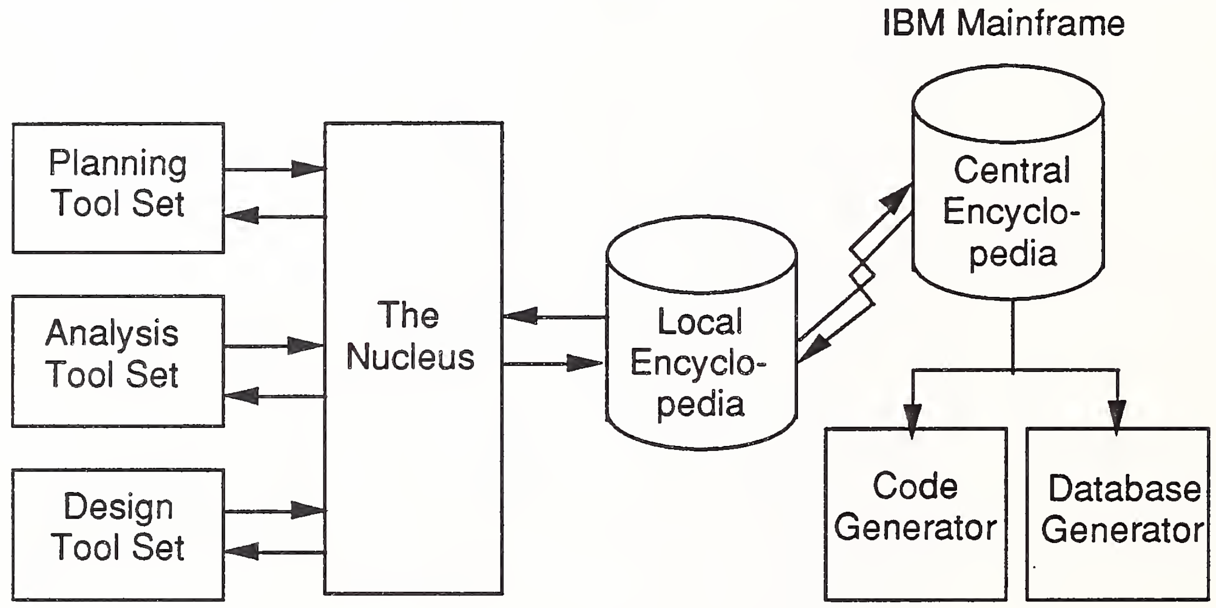
Revenue is expected to reach $30 million in the 1988/89 financial year and expansion is expected to continue at a substantial rate of 40-50% per annum on average over the next 5 years.

2. Systems Engineering Products

Exhibit IV-11 illustrates the architecture of the IEF. It should be stated at once that a key characteristic of the James Martin offering is that the methodology and the tool set are symbiotic. The use of the methodology implies and requires the support of an integrated CASE tool set.

EXHIBIT IV-11

STRUCTURE OF JAMES MARTIN ASSOCIATES' IEF



On the other hand, the design of the tool set is such that the comprehensive nature of the methodology is fully implemented and the user is not expected to need recourse to any other methodology or tools. This is a complete cradle-to-grave approach.

We have said that CASE concepts appear to go in threes. IEM/IEF is no exception. The methodology works with three levels of architecture:

- The information architecture describes how the business functions use the data and the information in the organisation. This architecture leads to scoping the work required for the next level of the architecture.
- The business systems architecture is the second level in the hierarchy and deals with what James Martin calls natural business system units. These are the business systems that any company would expect to

have, although in individual companies the boundaries may vary. Business systems support a set of business functions that in turn maintain a set of data. The systems are classified according to the level at which they work:

- operational
- planning
- strategy
- or a mixture of the above

- The technical architecture deals with the physical level of the systems, the equipment and the software required.

The IEM operates through seven levels of task and seven stages of a project to produce a working system. Key aspects of the approach are:

- The business needs are paramount.

- Postpone implementation decisions until absolutely the last minute.

- Capture all the needs within the organisation in order to be able to analyse the mutual impacts between systems in different business areas.

In summary, we can say that though the methodology is complete in itself it is not rigidly enforced. It is possible to break out of the methodology to use nonautomated features during certain phases of the life cycle and to bolt on the use of other tools. INPUT would classify the IEM methodology as therefore in the medium-strict category.

3. Marketing

The IEF is sold for upwards of $200,000, and a typical installation would come nearer to $800,000. The product is sold with the methodology, consulting, training and necessary ongoing support.

The company intends to establish itself as a world leader in CASE-integrated tool sets, and with this in mind has invested in a worldwide sales network.

The joint agreement with Texas Instruments is such that outside North America James Martin has the world exclusive rights for sales of the IEM and the IEF. Currently the product is aimed at blue chip companies, over 100 of which are in the present user base.

Looking further to the future, INPUT sees this company maturing as an excellency centre consultancy. There are questions about how it will cope with the expanding market for CASE tools in the medium-size and the smaller levels of company. The company is currently looking at a UNIX-based implementation of the system.

H Pandata

1. Company Background

Pandata was founded in 1970 by a consortium of organisations, which included:

- The Netherlands PTT
- Akzo, the Dutch chemical concern
- Gemini, the U.S software house later acquired by Sogeti

Since 1975 Pandata has been a member of the Cap Gemini Sogeti Group. It now employs 900 people and in 1987 had sales revenues of $50 million.

Pandata business areas are four in number :

- Consultancy
- Training and education
- Systems production
- Software products

In 1974 Pandata produced SDM (System Development Methodology). This was one of the reasons for the foundation of the organisation, since the original members needed formal methods for their internal systems work.

SDM is the most widely used methodology in the Netherlands and is also used to a more limited extent in Belgium, Scandinavia and West Germany. It is public-domain information and is sold as a book, tens of thousands of copies of which are now in circulation. SDM is taught in Dutch schools. A survey of DP installations in the Netherlands estimated that 50% used a formal methodology of some sort, and of these 60% were using SDM. The methodology has therefore 30% penetration, at least in the Dutch market.

2. Systems Engineering Products

In 1976 Pandata entered the training market for SDM and included it along with other methodologies in the Pandata Informatics Institute (PII) syllabus.

SDM covers a number of aspects:

- Control of the work
- The organisational framework
- The system life cycle phases
- Manuals
- Reports

SDM makes a keen distinction between the overall methodological framework and the techniques used in implementing specific activities within the life cycle. A number of techniques are supported, including:

- Precedence diagrams
- Data flow diagrams
- Data structure diagrams
- Decomposition diagrams
- Relation-matrics (BSP)
- Information structure diagrams
- N2 charts
- Flowcharts
- Jackson diagrams
- Nassi-Schneidermann diagrams

In 1985 Pandata decided to enter the CASE tools market with a product called the System Development Workbench. One of the basic aims of this market entry was to preserve its existing training revenue stream, since it was envisaged that without the software tools to support the methodology, the training activity would be erroded and overtaken by other suppliers.

The System Development Workbench (SDW) is a PC-based analyst tool operating in an open architecture framework. It is a modular system with the modules tied to each phase of the system life cycle.

Over 100 analytical design techniques are supported. These diagramming techniques are supported using the PC's graphical capabilities. An example of a data modelling diagramming technique is given in Exhibit IV-12, which shows the data modelling structure of the SDW's system encyclopaedia.

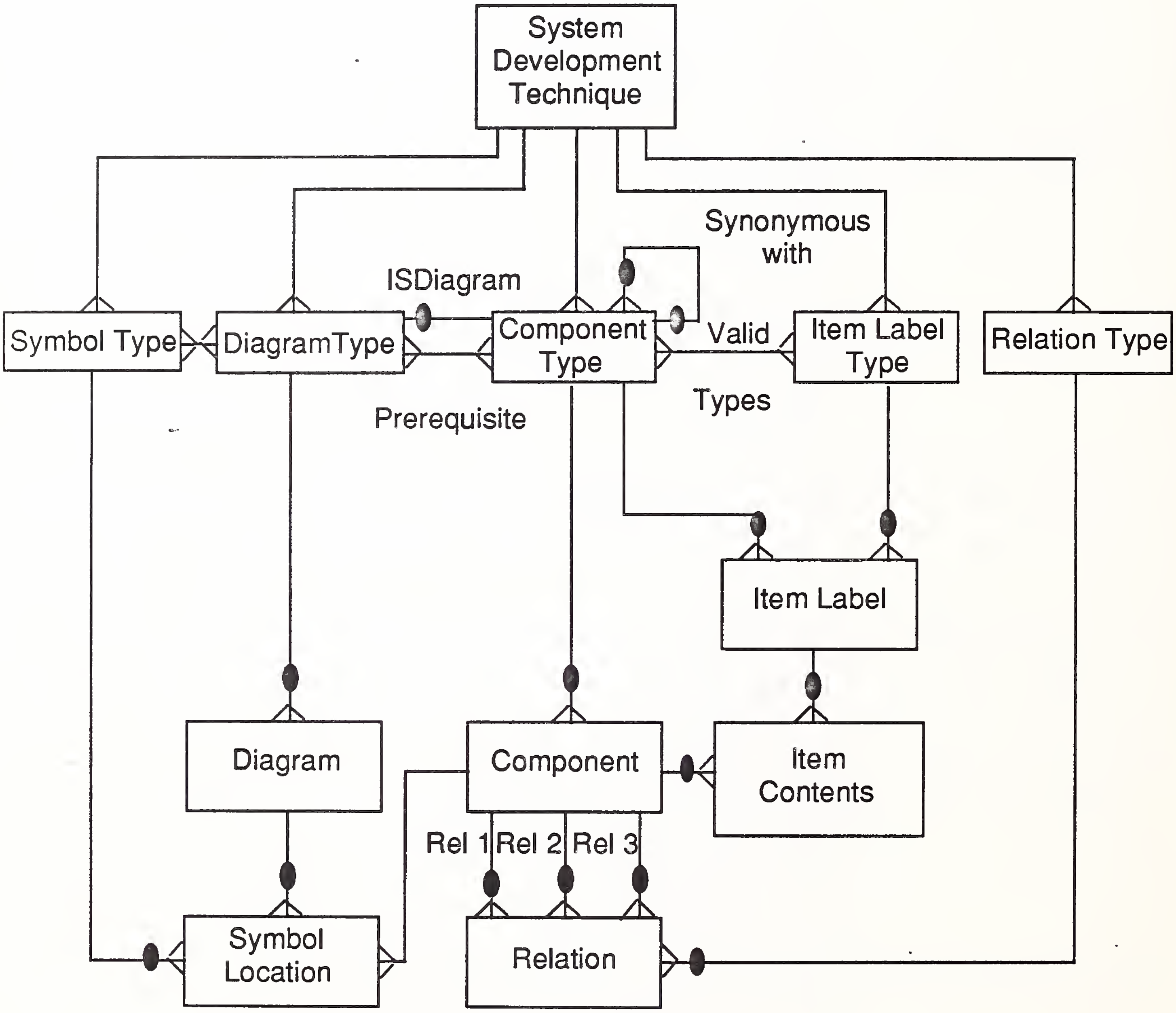
Pandata believes that flexibility is of the essence in providing a good analysis workbench. flexibility must be accompanied by discipline to tailor the methodology to the chosen techniques and standards. With this in mind the workbench incorporates a metamodel module that allows for the set-up of the techniques and standards to be used on an individual project at the start of a project.

There are currently five modules available in the workbench:

- SDW— DM for data modelling
- SDW—IA for information analysis
- SDW—DA for data analysis
- SDW—FA for function analysis
- SDW—FC for flow charting

EXHIBIT IV-12

THE STRUCTURE OF THE SYSTEM ENCYCLOPEDIA IN PANDATA'S SDW WORKBENCH



Other modules currently under development cover the following functions—prototyping, structure design, multiple workstation use, and certain specific diagramming techniques.

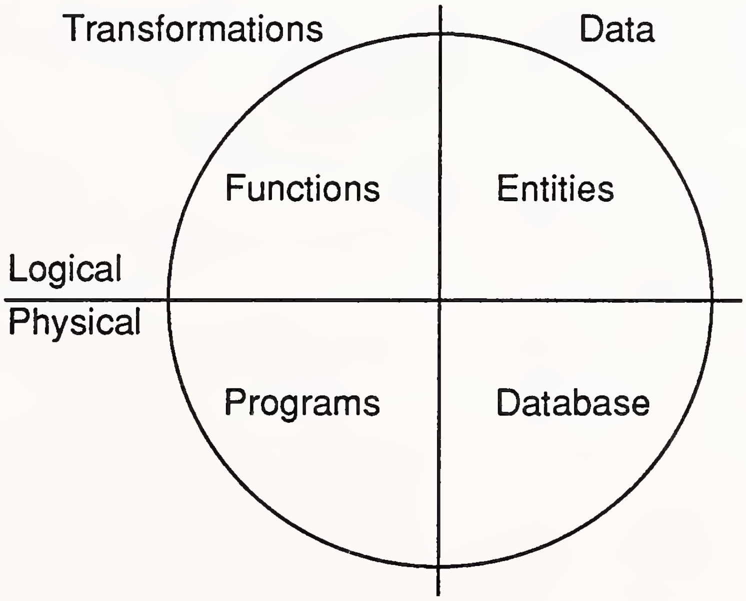
In 1988 connections to code generation modules and database managers are planned.

Pandata started with the analyst functions in the workbench because most faults are traceable to the analytical phase and because the cost of fixing faults from this phase is most costly.

The implementation architecture of the workbench assumes that it operates within a source environment based on the PC. The target machine for which the system is to be developed is considered to be a separate environment. The architecture assumes the use of a modern data dictionary, whose structure is illustrated in Exhibit IV-13.

EXHIBIT IV-13

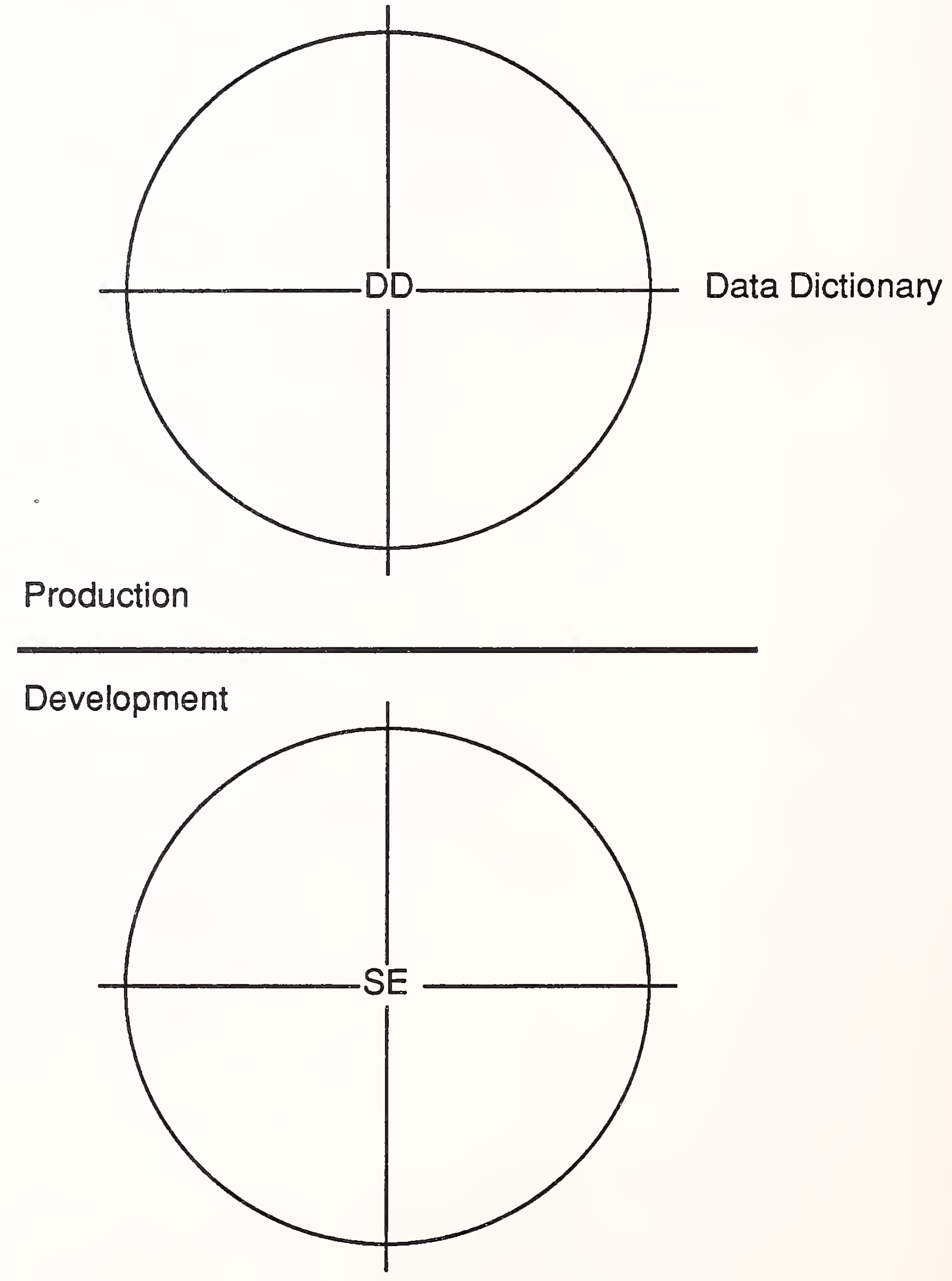
ARCHITECTURE OF MODERN SYSTEMS: DATABASE WITH ACTIVE AND INTEGRATED DATA DICTIONARY



The separation between the source and target environments is illustrated in Exhibit IV- 14, which shows the difference between the data dictionary in the production environment and the freely definable system encyclopaedia in the development environment. INPUT strongly supports this architectural split because it permits easier evolution of methods in a real-world environment. The data dictionary handles today's production system descriptions; the system encyclopaedia can handle tomorrow's technology. The two can overlap and can be made compatible via an interface module.

EXHIBIT IV-14

SEPARATION OF PRODUCTION AND DEVELOPMENT AREAS IN THE PANDATA SDW



Freely Definable System Encyclopedia

Although the workbench is currently implemented in a single-user version, but multiple-workstation and multiple-user versions are envisaged, in which split or shared system encyclopaedias are integral to the design.

The workbench is currently implemented in a single-user version, but multiple-workstation and multiple-users versions are envisaged, in which split or shared system encyclopaedias are integral to the design.

The workbench is implemented in both the old PC and the new PS/2 systems. A decision has been taken to support the OS/2 operating system, and it is envisaged that UNIX will also be implemented later.

The workbench is built for use within a distributed database environment. In this, Pandata's thinking is ahead of the majority of the other workbench product companies in the market-place today. Pandata believes that the move towards distributed data processing requires greater control of information conventions within the user organisation. In practice greater control means using a controlled set of common data definitions. Without these data, ownership and control of a corporate database is impossible.

3. Marketing

The workbench is currently sold as a free-standing software product backed by the company's support and maintenance facilities. Although now only installed in single-user units, the move towards the multistation environment is envisaged for 1988.

Approximately 2,500 modules have been sold, including those supporting Pandata staff's software projects. On average a user will take between three and four modules of the workbench, the first of which always includes the system encyclopaedia maintenance module. Users can start with one or two modules and work up from there. Two clients have installed hundreds. The user base in mid-1988 was approximately 250.

The market is envisaged to be large, with some 30,000 analysts and designers in the Netherlands to be covered. The target for 1988 is to sell 300 workbench modules; the campaign is currently well on target. Main competitors are seen as the Excelerator product and Arthur Young's Information Engineering Workbench (IEW).

Future markets are seen as including West Germany and France (if the French Board of CGS accepts the product for wider marketing using its group resources).

Pandata has formed alliances with other professional services vendors in Holland that can provide other facilities for overall life cycle support. An example is the COBOL Generator from DSA, to which an interface to SDW has been built.

The technical orientation of the SDW product makes it eminently suitable for sale within the U.S market, where a tool-oriented modular approach would be appreciated. Within its own domestic market Pandata will need to put more sales effort into the programme as rival products come on-stream.

I

Other Suppliers' Systems

Mention should also be made of the following methodologies and techniques in current use and encountered during INPUT research:

- BIS MODUS, a methodology adopted by the BIS software house for internal use and also sold as a methodology to the client base

- GANE & SARSON, a diagramming technique for data modelling and flow modelling, which is in common use in a number of methodologic systems

- ISAC, a DP methodology taught in Scandinavia

- SADT, a methodology favoured for NATO contracts

- MASCOT3, a project support methodology designed at the RRSE (Royal Radar and Signals Establishment) in the U.K.

- CORE, a U.K. Defence department procurement specification methodology

There are also several methodologies designed by professional services vendors initially for their own use, but now implemented in client projects unless the client stipulates another methodology:

- HOSKYNS' PRISM
- CGS's EXPERT
- CSC's DSDM

V

Current Vendors of Software Tools

V

Current Vendors of Software Tools

A

The Sector Overview

Exhibit V-1 introduces the concept of the life cycle footprint. This footprint is analagous to the system footprint found in the Yourdon Structured Analysis Method (YSM). However, it concentrates on the footprint of the problem the computer system is addressing, whereas the system footprint of YSM concentrates on the characteristics of the solutions embodied in the system.

On the three axes, we have the three independent variables of importance: On the X-axis we have USAGE, defined as the amount of exposure to hardware a piece of software will encounter in its lifetime and defined to include the criticality to human life of that usage. Three aspects should be considered:

- The number of licences or copies of a piece of software
- The number of runs a piece of software will undergo in its life, this being a function of the run frequency
- The criticality or safety factor involved

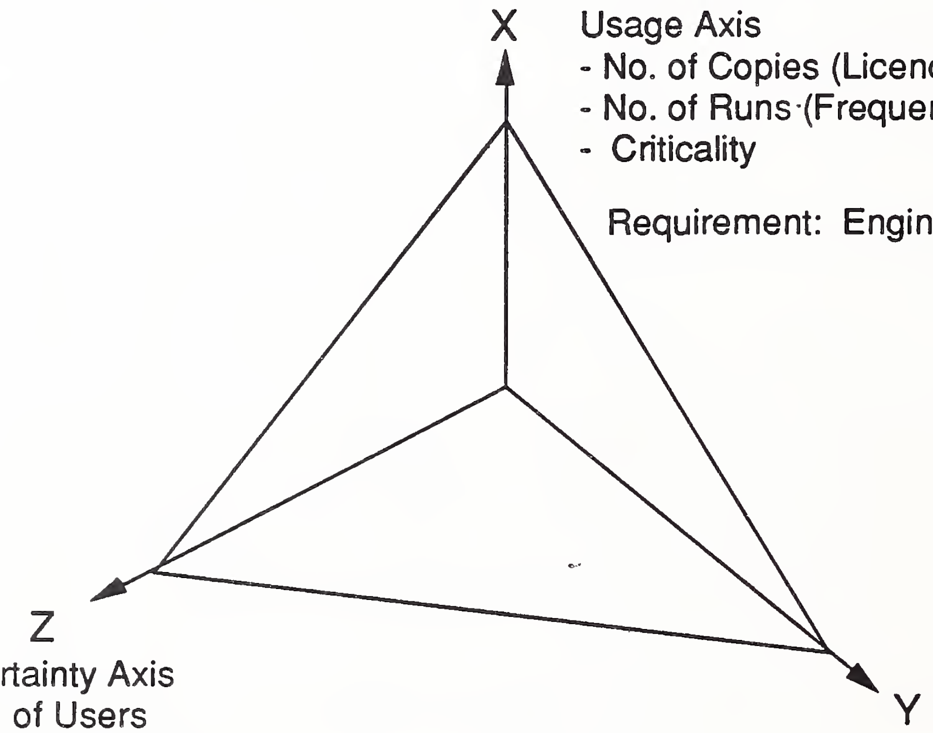
On the Y Axis we find COMPLEXITY. This relates to the amount of functionality needed in the system as well as the structure likely to be imposed. A third aspect contributing here is the hardware platform upon which the system is to stand.

On the Z-Axis we have the usually forgotten characteristic of UNCERTAINTY. The uncertainty in the definition of the problem will relate to a number of factors:

- The number of users who will need to be involved in its specification and/or use

EXHIBIT V-1

THE LIFE CYCLE FOOTPRINT



Usage Axis
- No. of Copies (Licences)
- No. of Runs (Frequency)
- Criticality

Requirement: Engineering Disciplines

Uncertainty Axis
- No. of Users
- Comprehension
- Effect on Organisation

Requirements:
Ground-Level Experience
Interpersonal Skills

Complexity Axis
- Amount of Function
- Structure
- Platform

Requirements:
Training
Expressiveness

- The comprehension its users have of the use of the system

- Most important of all, the effect on the organisation of implementing the system—the feedback effect

The engineering style discipline implied in the CASE tool market will increase with the amount of usage to which a system product would be put. In other words, if the life cycle footprint has a high point up the X-axis indicating that it is going to be a widely used product (for example, a piece of system software) or is to be run with a high frequencey (e.g., a daily on-line transaction processing system), then it must be rigorously engineered in the same way that any piece of electrical or mechanical equipment must. This requirement is less important if, for example, the product is a small one-off PC-based system for a single user.

The complexity axis is likely to have a high value in technical and scientific systems where the definition of the system requirement may take a considerable time and will probably require the prototyping of routines and the exercising of critical sensor-driven modules in simulated, close-to-real-life conditions.

It is the uncertainty component in the life cycle that is often ignored. This omission leads to the definition of systems that are not what the user wants. Also it leads to higher expectations than can be fulfilled. This gap is bridged only by the use of experience on the part of the system developers and the high quality of their interpersonal skills.

Exhibit V-2 illustrates the modern-day system development environment within an organisation of any size. Systems are required at four levels, starting with the PC at the personal level and working through to the IS department responsible for the corporate data resource. How vendors should react when selling to this environment will depend upon the life cycle footprint of the problems addressed at each level. The objective of this chapter is to allow vendors to derive a life cycle footprint for their particular product areas and to be able to review their sales and marketing strategies accordingly.

This chapter will also review current developments in the existing market segments and will encourage vendors to promote products in ways appropriate to the life cycle footprint of their choosen target systems.

B

Mainstream Languages and Database Environments

The industry is now poised somewhere between the third and the fifth generation of software development, taking facilities from a range of different approaches characterising each generation. In future, one generation will not dominate to the exclusion of other earlier ones. CASE will assist in subsuming each passing generation into the substance of the next.

Fourth-generation languages (4GLs) were brought to market in order to speed up program production. Their two aims are:

- To speed program development

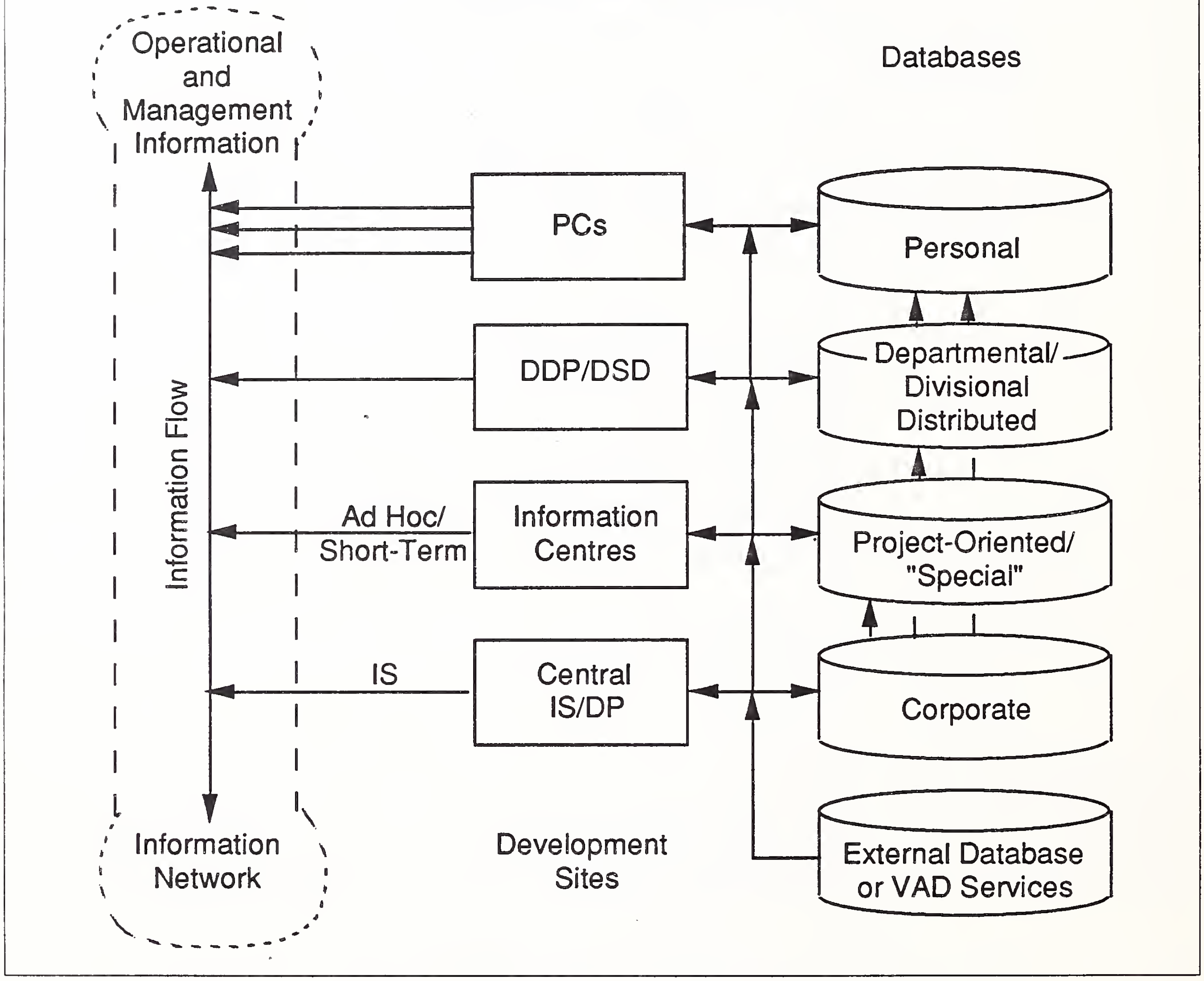
- To satisfy the user's requirement sooner, in order to forstall the inevitable changes to specification that users bring as time passes

The techniques of fourth-generation languages are:

- A screen-based dialogue orientation

EXHIBIT V-2

THE ENVIRONMENT OF DISTRIBUTED SYSTEMS DEVELOPMENT (DSD)



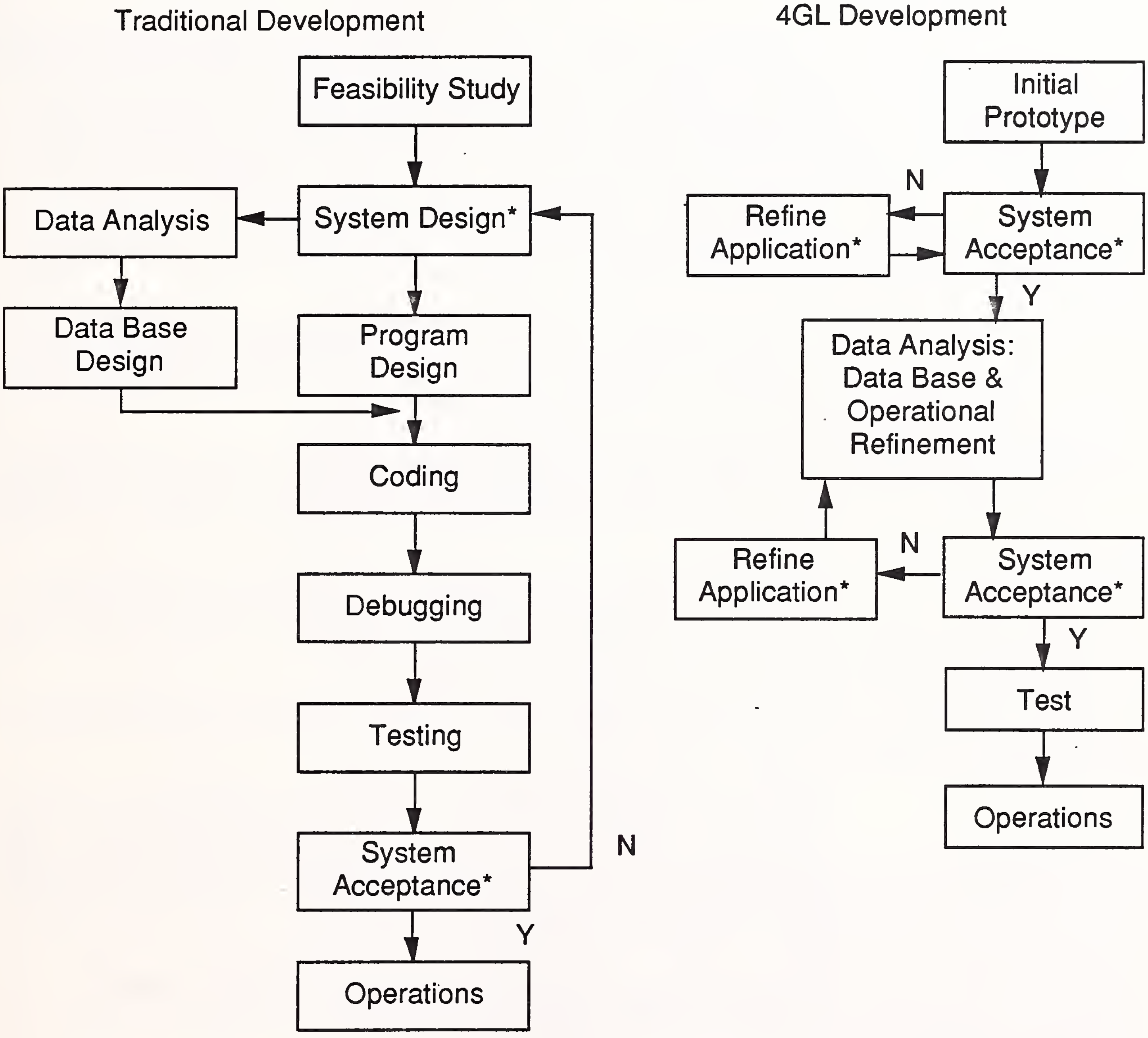
- Specification via the "prototyping" approach

- Data-oriented thinking on the part of user and developer (as opposed to the traditional function-oriented approach taught to professional development staff up to and including the third-generation of system languages)

Exhibit V-3 illustrates the methodological approaches of the traditional development method and that of a 4GL.

EXHIBIT V-3

4GLS' ATTEMPT TO CORRECT FLAWS IN TRADITIONAL SYSTEMS DEVELOPMENT



*Significant User Involvement

What has in fact happened is that three types of 4GLs have grown up :

- The micro-based 4GLs are really extensions or attachments to generic packages. For example, the dBase family and Lotus 1-2-3 both have linguistic add-ons. Many people think of these as packages rather than languages.

- The traditional 4GLs are the descendants of the early report generators, which were invented as long ago as the 1960s to allow for the extraction of reports on a routine or an *ad hoc* basis from files of any type. Members of this development stream are FOCUS, RAMIS and NOMAD. This type of 4GL serves the ad hoc information centre (IC) applications and the decision support software (DSS) sector. The majority of these languages grew up as interpretive in operation, although in many cases compiler-based versions are now available.

- The mainstream development of 4GLs has become database orientated, and the languages are often associated with well-known database products. These are now called Developer 4GLs. Here again there is a dividing of development streams:

- ADR (now part of Computer Associates), CULLINET, CINCOM and PANSOPHIC are mainframe system software companies with database products. All have moved into the 4GL business.

- COGNOS, ORACLE, INGRES are the main contenders for similar products for the DEC VAX market, although they have in some cases also provided IBM mainframe products.

- A newer set of suppliers are coming forward with UNIX-based products of a 4GL nature.

This 4GL development stream has taken place in parallel with the move to adopting the relational database as the standard for the industry. This move has shifted the focus from the pure screen/report orientation of the original developer 4GLs to include the “relation” as one of the principal concepts of the system developer.

The drawbacks that have become apparent as 4GLs have spread in acceptance throughout the industry are:

- There has been a proliferation of offerings without any standard definition of what a 4GL is and without any standard language definition.

- Following this there has been a proliferation of specialist programmers and analysts with knowledge of one or more of these languages and their intricacies. This has had the effect of worsening the shortage of professional system developers throughout the industry.

- There are performance-related charges that can be levelled against the 4GL.

- The 4GL development concept blurs the boundary between the design of the system and its production and in doing so it tends to destroy the professional engineering approach, which is so necessary to the production of quality software.

The latest generation of 4GLs is now coming to market with the label of 4th-Generation Environments (4GEs). They aim to be general purpose in that they equally well suit the in-house end user wishing to write a small system and the professional developer from the in-house DP department or from an external systems house or value-added reseller (VAR).

Exhibit V-4 shows the structure of the SEACHANGE 4GE, which has been on the market for less than two years and is typical of this new breed. Like many of the recent products brought to market, this is UNIX- and C-based.

Language products from COGNOS (Powerhouse), CULLINET (KnowledgeBUILD) and Oracle are all of the 4GE type offering fill-in-the-blanks and template programming. Another trend is to integrate the 4GE with an expert system. Companies such as CULLINET (with Application Expert) and Parsoft (with TODAY) are examples of this type.

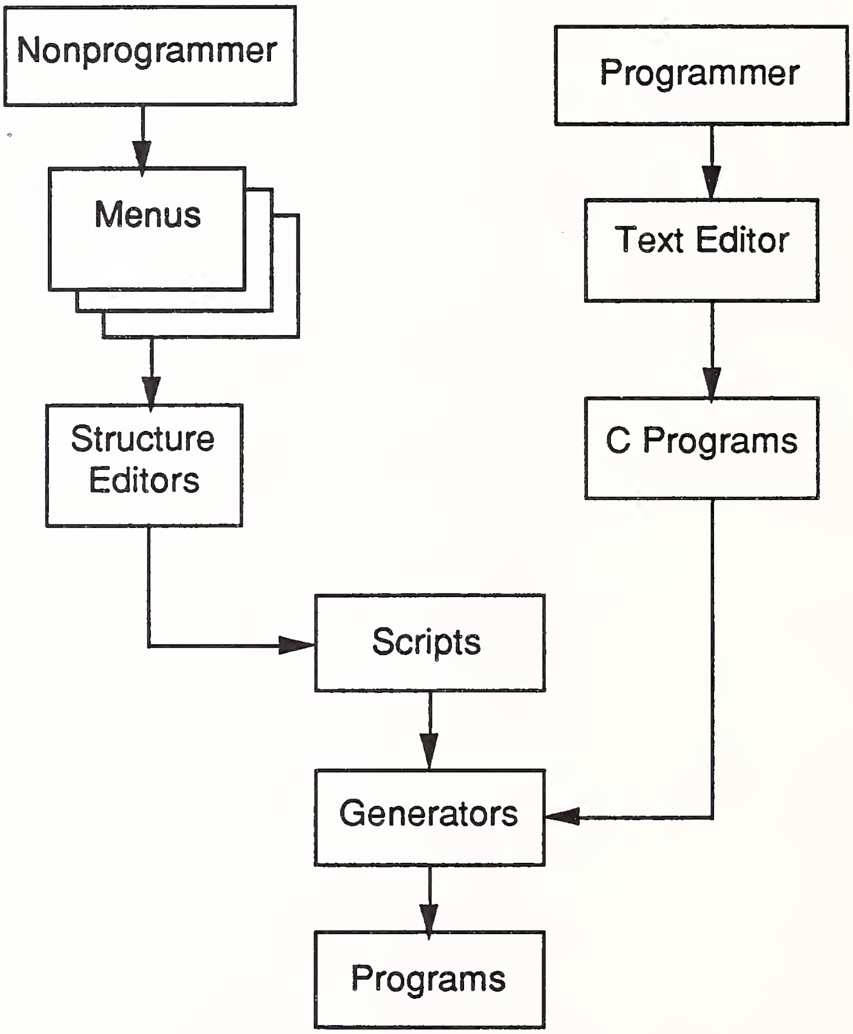
To overcome some of the interproduct compatibility problems, many product developers are now adopting SQL (Structured Query Language) as the standard intermediate language, not only between user and database but also between relational and nonrelational databases, databases and knowledge bases, and database and object-oriented programming methods. This approach has been sanctified by IBM's strategic choice of DB2 as its future database flagship.

The problem of how to set 4GLs into the framework of a methodology remains. 4GLs cut across the traditional boundaries. This problem raises the question of whether they are equally well suited to the professional developer and to the end user. The best experience from large organisations shows that as soon as a system involves the use of existing corporate or departmental databases or the design of specific data structures or any form of data complexity, one needs the professional data processing specialist. Even if end users are capable of doing the work themselves, programming will take end users away from what they should be concentrating on. Calling the work by some other name than programming doesn't change this fact.

INPUT recommends most firmly that vendors and users alike must build on the advantages of having the professional intermediary. Constantly

EXHIBIT V-4

STRUCTURE OF A 4GL FOR PROGRAMMERS AND END USERS



trying to do away with intermediaries is a case of the industry shooting itself in the foot. It is immaterial whether one calls the intermediary a programmer, an analyst programmer, an analyst or a consultant. At all levels of system development the intermediary has a place as soon as the system rises above the level of the most trival. The trivial-to-serious threshold is crossed as soon as shared data are involved.

The latest development in the mainstream language area has been the maturing of application generators. Though this class of product has been around for several years, it is only recently that systems have developed with:

- Standard interfaces to front-end CASE tools
- The correct level of performance from the generated target code

These products, especially mainframe-based offerings such as PANSOPHIC's TELON and SAGE's APS, have aroused a considerable reawakening of interest in the COBOL language.

The use of mainframe-based application generators cuts across the trend towards separating the development environment from the target environment. The long-term trend will position application generators in the production side of the system development life cycle and has implications for the development of standard data dictionaries and standard system development databases (sometimes called encyclopaedias or repositories).

INPUT favours the separation of the development environment from the production environment in order to strengthen the professional status of the system developer by giving the developer not only the tools but also the time and other resources necessary to make a task efficient. Too often in the past the development staff has had to take a secondary place in obtaining machine resources to achieve objectives.

The development machine environment can be standalone PC based, LAN-based or UNIX workstation based. The migration of development units, from traditional methods to the use of more rigorous methodologies supported by CASE tools, will be made smoother by the adoption of the separate development machine environment, although this migration must be linked effectively to the target environment.

This separation is already "standard" in the microprocessor development field, and in the CAD/CAM world it would be unthinkable to design something on the same platform on which it was going to be produced.

C
Standalone CASE Tools

CASE started as a sector in which free-standing unit tools were made available for supporting various system development functions:

- Analysis
- Design
- Program Generation
- Test Aids
- Etc.

Different suppliers have developed different strengths depending upon the background from which their products have sprung.

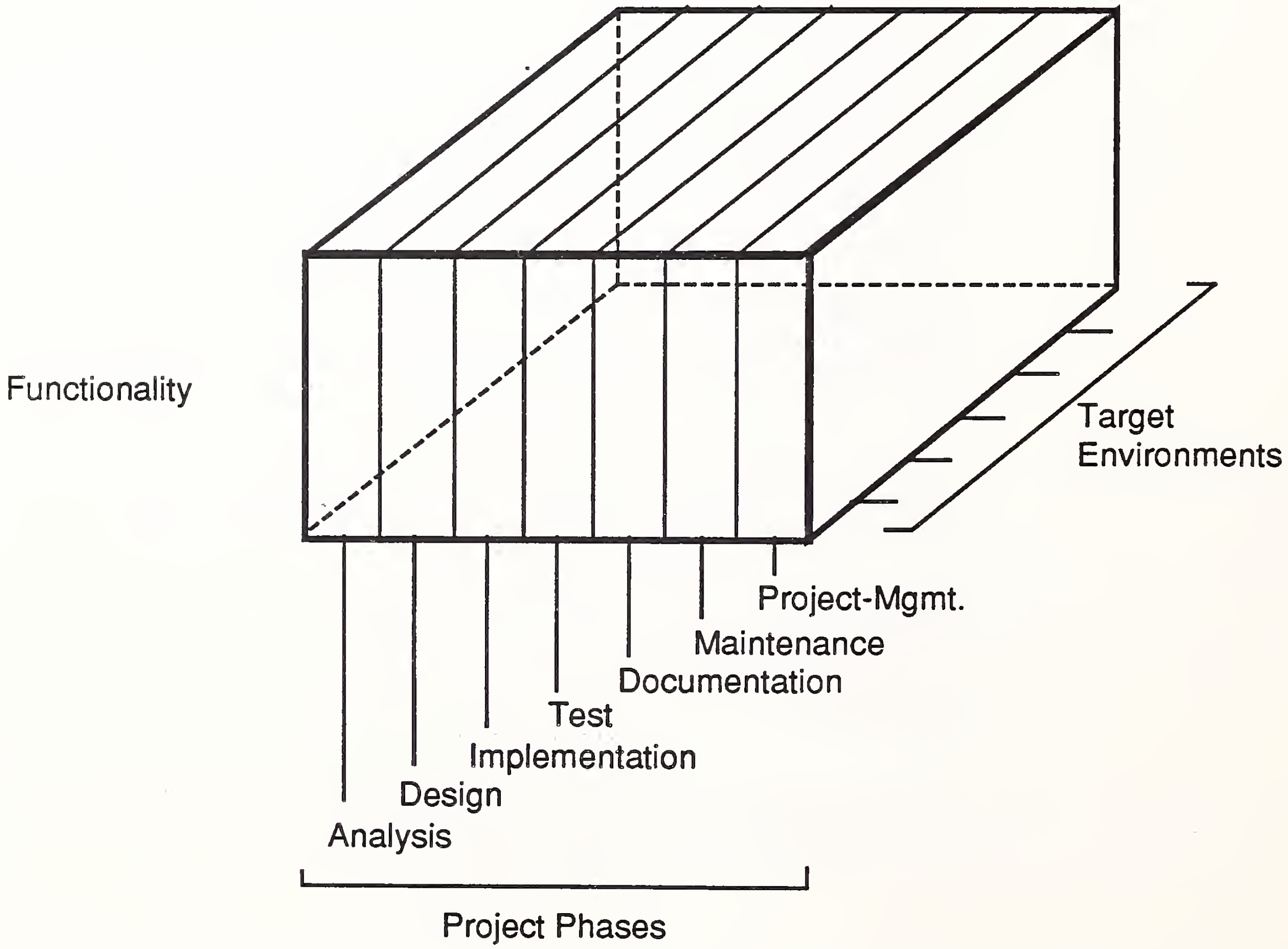
The second-generation of products now reaching the market addresses the connectivity issues:

- Interfacing to other tools
- Working alongside
- Communicating between team members and to an overall management system

A Cartesian view of the CASE tools sector in a life cycle landscape is given in Exhibit V-5. This exhibit shows the three dimensions: phasing of projects, functions of tools and target environments.

EXHIBIT V-5

BUILDING-BRICK APPROACH TO SATISFYING CASE REQUIREMENTS

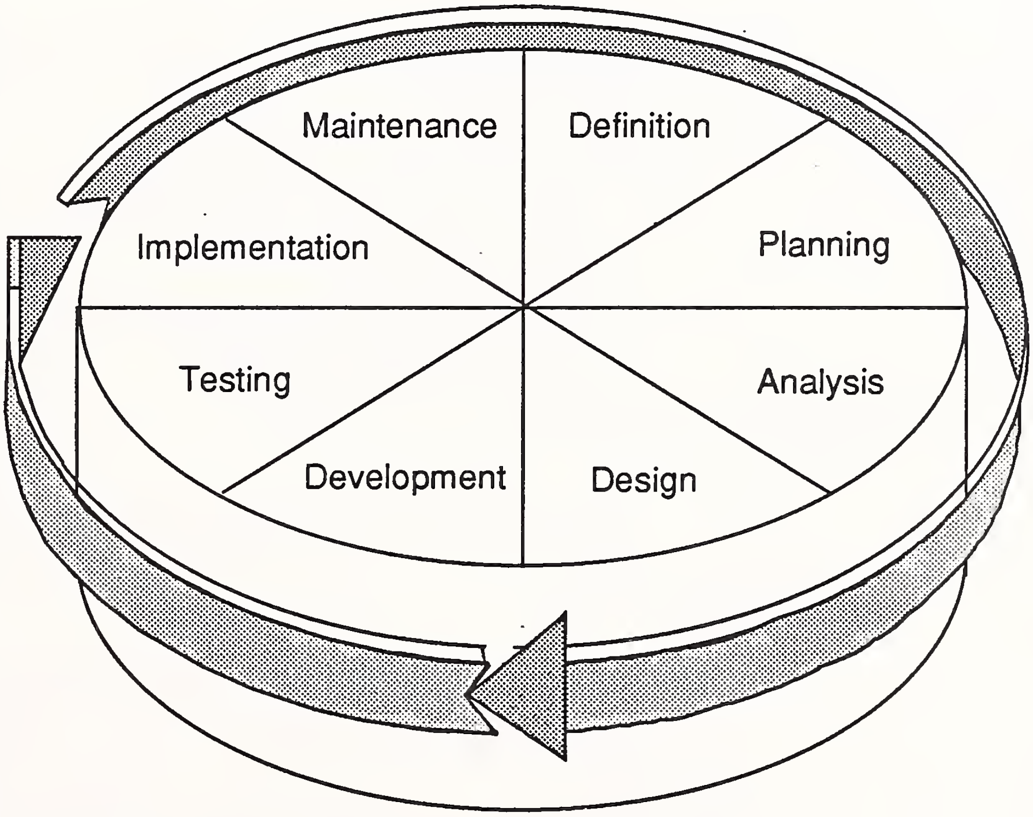


This chart emphasises the building-brick approach to CASE, but says nothing about the connectivity or interoperability between them. It tends to imply the 'waterfall' or start-to-finish approach to system development.

By contrast, Exhibit V-6 shows the life cycle as a revolving wheel, thus emphasising the interworking requirement and the continuity of wraparound between Implementation, Maintenance, System Redefinition, and Enhancement or Rewrite.

EXHIBIT V-6

THE SYSTEM LIFE-CYCLE SEEN AS A REVOLVING WHEEL



1. Front-End Tools

Standalone tools are usually classified into front-end and back-end (this is sometimes called Upper and Lower CASE).

Front-end refers to the analysis and design; back-end to the code, test, maintenance etc. We have already seen how difficult it can be to separate these two areas; there is a constant tendency to want to collapse them into one phase in order to simplify or shorten development time. The diagram in Exhibit V-7 illustrates the connections between the initial design phase and the synthesis of the constructed system to make an integral working whole. The crossover points are effective at the testing and verification stages.

Front-end tools have started at the initial analysis stage because the better the new requirements specification, the better the quality of the final system and the sooner the end user will be satisfied with the product.

Two types of vendor have emerged:

- Those with method-independent tools
- Those with method-dependent tools

Leading vendors of front-end products (often called analyst workbenches) are:

- Index Technology with Excelerator
- Arthur Young with the IEW (Integrated Engineering Workbench)
- LDMS with Auto-Mate Plus
- Pandata with SDW
- McDonnell Douglas with ProKit*Workbench
- GEI with Promod
- LOGICA with MacCadd
- Teamwork

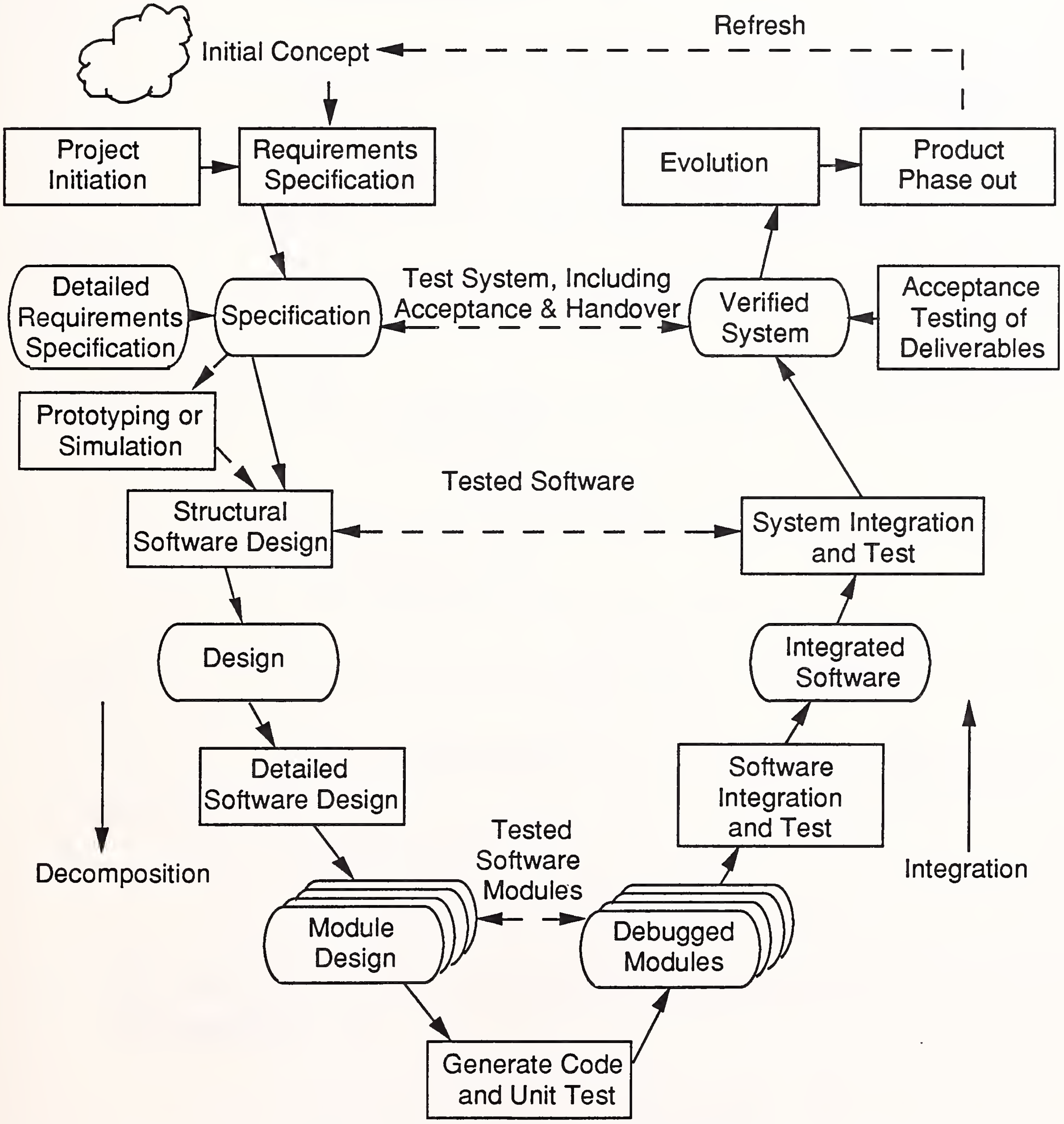
There are also many CASE tool sets that claim to be integrated, some of whose modules perform similar sets of functions to those described above.

Excelerator sets the pace in this market sector with its current number of licences exceeding 13,000 worldwide (2,000 or more in Western Europe). Its facilities include:

- Support for a variety of diagramming techniques
- Data modelling with verification and consistency checks
- Screen and report design
- Dictionary maintenance
- Specification documentation preparation
- Data sharing for team projects
- Interfaces to other well known products

EXHIBIT V-7

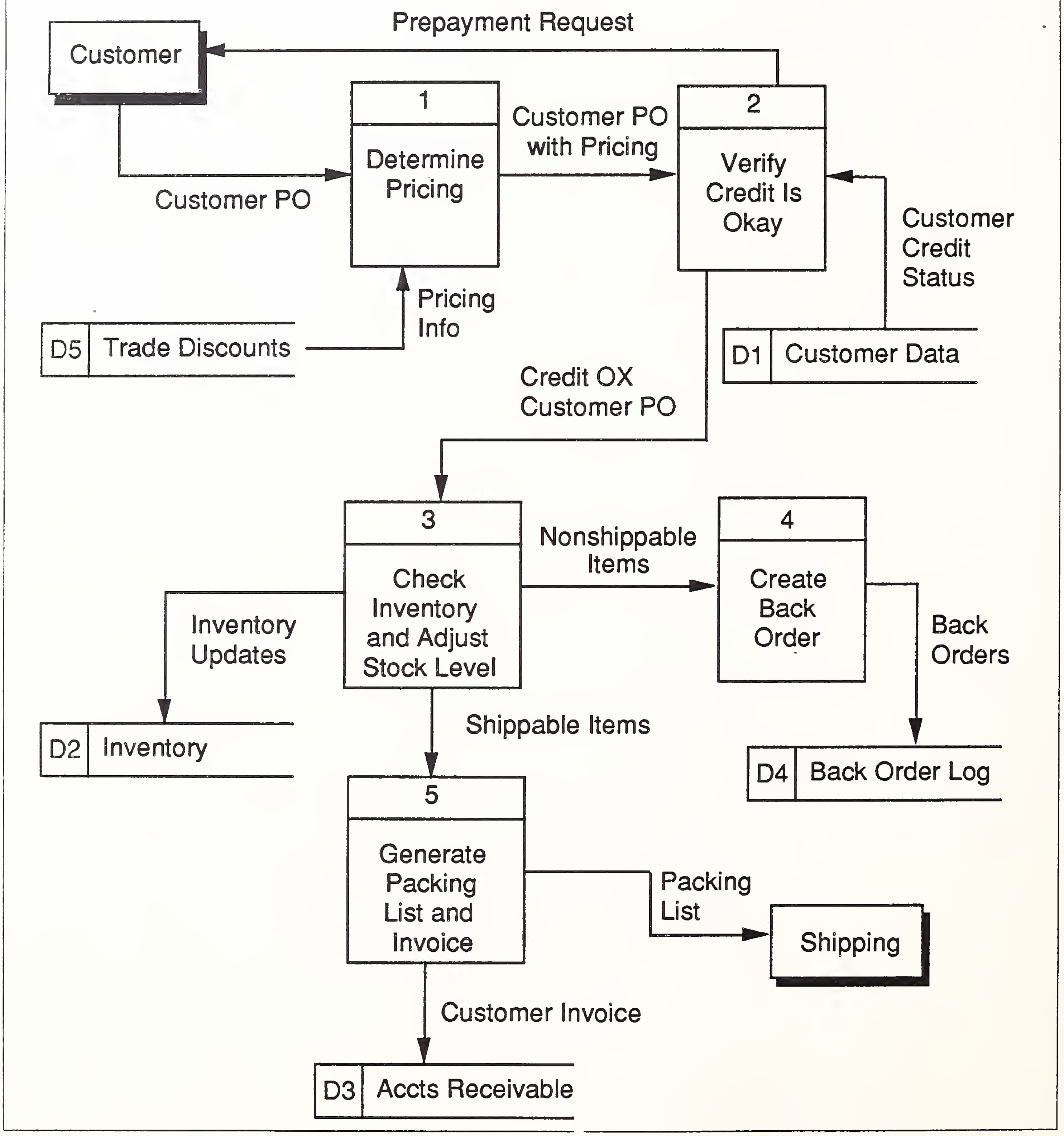
THE SOFTWARE LIFE CYCLE SHOWING DECOMPOSITION (ANALYSIS) & INTEGRATION (SYNTHESIS)


Refresh
Initial Concept
Project Initiation
Requirements Specification
Evolution
Product Phase out
Detailed Requirements Specification
Specification
Test System, Including Acceptance & Handover
Verified System
Acceptance Testing of Deliverables
Prototyping or Simulation
Tested Software
Structural Software Design
System Integration and Test
Design
Integrated Software
Detailed Software Design
Software Integration and Test
Decomposition
Tested Software Modules
Integration
Module Design
Debugged Modules
Generate Code and Unit Test

The example of a Gane & Sarson diagram, as shown in Exhibit V-8, is typical of the diagramming support given by this product.

EXHIBIT V-8

A GANE & SARSON DATAFLOW DIAGRAM



Over 40 suppliers operate in the front-end sector with 60 or more products. The emphasis is in different parts of the life cycle, stretching between requirements definition and detailed program design. One common thread to all these tools is the use of graphics techniques to implement modern database design methods.

2. Back-End Tools

This is an area less easily associated in the industry mind with CASE, since until recently it consisted of the traditional Compilers, Assemblers and Testing Tools, all of which have historically been part of this subsector before the name CASE was coined.

The recent advent of the mainframe application generators (after being around in one form or another for many years and rejected because of poor performance) has added to the growth of the CASE sector in its own right. Several vendors in this sector claim that they are in the Integrated CASE sector (I-CASE, described later in this chapter).

Major players in this sector are:

- PANSOPHIC with TELON, an IBM-mainframe-based product

- SAGE with APS, also for the mainframe market

- CGI with PACBASE, now built into an integrated product with a workbench front end

- MICHAEL JACKSON with its code generators

- The CORTEX Application Factory, now likely to be replaced by the company's Corvision System

There are a few small specialist players concentrating on neglected areas in the life cycle, such as testing and documentation:

- Structure Analysers
- Specification and Verification Analysers
- Static and Dynamic program analysers
- Configuration and Change Management systems

The traditional area of project management continues and is being taken up into the overall CASE sector with products specific to system engineering:

- Hoskyns' PC-based product PMW, which is often interfaced to by other vendors' tools.

D AI Techniques

Artifical Intelligence (AI) was originally thought of as the basis for the fifth generation of software development. It was thought to be able to replace earlier methods based on procedural languages. In practice AI systems have been carving a number of niches:

- The supply of specialist facilities such as natural language translation.

- Embedded in many products in varying degrees, we now find AI techniques as an enabling technology.

1. Expert Systems

Expert systems have been established in many large organisations on a pilot basis. They have been put under the control of specialist expert systems groups separate from the central IS function, and are now found to be performing useful functions in a range of industries on a relatively small number of application types:

- Maintenance
- Diagnosis, e.g., medical
- System and configuration management
- Business rules

Essentially expert systems represent a tool to support complex decision making where technical data and technical judgement are required. The characteristics of the life cycle footprint of the expert system are:

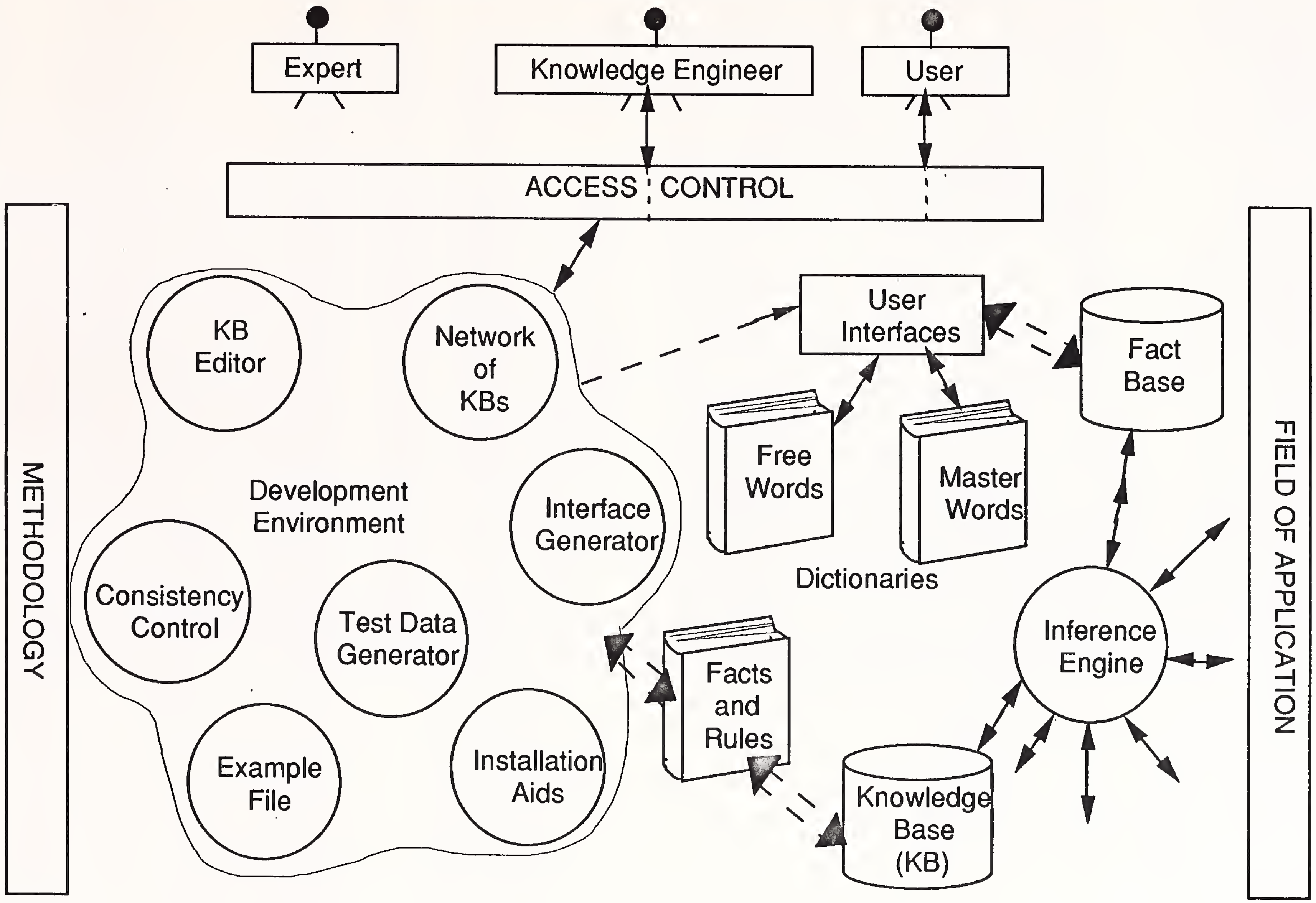
- It has medium to high complexity
- It has low to medium usage
- It is sometimes of high criticality
- Uncertainty is low

One of the early aims of AI was to be able to handle problems in which a degree of uncertainty existed. It has in practice been found that the majority of expert systems in commercial or industrial use today have resulted from good definition of problems without attempts to incorporate techniques for handling uncertainity, over the details of which specialists argue strongly. It also been found that the organisational impact is one of enskilling rather than deskilling the function AI assists.

Exhibit V-9 illustrates the complexity of the development environment within which expert systems operate in medium-sized and large organisations:

EXHIBIT V-9

ELEMENTS OF AN EXPERT SYSTEM DEVELOPMENT ENVIRONMENT



- The knowledge acquisition problem is capable of reasonably cost-effective management. Experience shows that the best knowledge engineers are generalists with good interpersonal skills. This discovery gets round the problem of whether to teach the knowledge engineer the expert's experience or on the other hand to make the expert into a computer professional. This is one more example of the DP professional getting recognition—and not too soon.

2. Prototyping

Many complex problems can only be solved after solution scenarios have been simulated in as near to real-life conditions as possible. This strategy is in line with the fact that many engineering problems start with the building of the prototype, e.g., in the case of cars, trains and aeroplanes. Computer simulation is now an essential step in many of these operations; simulations are major projects in themselves. The use of simulation or prototype building to the stage where the measurement of performance can be made is not normally affordable for a commercial DP system but is absolutely essential for safety-critical systems.

With the techniques of objective-orientated programming, frame- and rule-based control system top-end AI vendors like IntelliCorp are active in the simulation business. IntelliCorp's KEE (Knowledge Engineering Environment) product now has over 2,500 licences in the world among over 500 major organisations, of which about a third are in Western Europe.

Exhibit V-10 illustrates the levels of problem solving at which AI-specific systems can operate:

- At the bottom layer the basic product development environment offers a low-level capability in, usually, LISP or PROLOG

- On top of this layer can be built certain generic tool kits specific to a range of applications, such as simulation kits and interfacing products

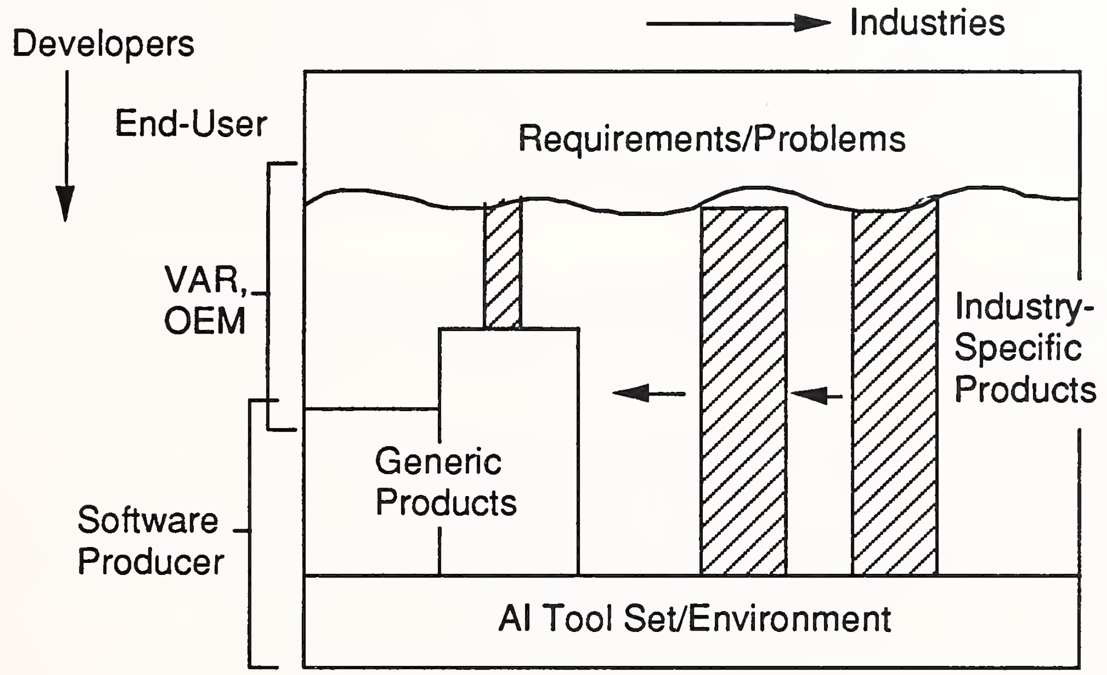
- Industry-specific products can be built on one or another of these platforms, either by the software producer, a value-added reseller or the end users themselves.

3. Bridging

The interfacing between incompatible CASE building bricks or between the run-time components of systems built with different language generations is an area to which the flexibility of AI techniques can bring considerable benefit.

EXHIBIT V-10

MARKET STRUCTURE FOR AN AI TOOL SET VENDOR



Examples of bridging are:

- NEXPERT is a run-time library allowing for access to many AI data structures and control mechanisms that can be used to add facilities to the calling program and/or to make the program integrable with non-AI commercial or technical systems. The product is available on four hardware platforms, ranging from the high-end PCs to VAX, including Apple Macintosh and UNIX workstations.

- KEEconnection, is an IntelliCorp product that aims to provide a combination of the run-time performance of a database with the expressiveness of a knowledge base by providing a software bridge between relational databases that use SQL as their query language and knowledge base applications developed using KEE. The product should allow for data to be maintained efficiently in one or more databases, but accessed as necessary by any standard KEE application.

4. Embedded AI

At the rule-based logic level, many CASE tools cannot do without the enabling AI technology:

- The encyclopaedia of James Martin's IEF

- The consistency checking of tools like LBMS's AutoMate Plus and Index Technology's Excelerator

- SAPIENS, an advanced system development language that looks like a 4GL but claims to be the only true 5GL in the market-place. To produce complete systems, SAPIENS uses a whole set of embedded knowledge about the structure of business applications and the IBM development environment. SAPIENS is an Israeli-built product marketed with Swiss investor money by SYSTOR-SAPIENS AG of Zurich.

At the frame-based, object-orientated level, most CASE tools are still at the R&D laboratory stage.

E

Comprehensive Environments (IPSE, I-CASE)

We can distinguish three generations of IPSE:

- The first is the Maestro-like IPSE, in which the construction of the software was the primary objective. These IPSEs are control and documentation orientated.

- The second generation of IPSE includes systems generally known as I-CASE and products such as PACBASE from CGI and QUICKBUILD WORKBENCH from ICL, as well as the James Martin Associates/TI/IEF. These systems are workbench orientated and usually aimed at the commercial systems market.

- The third generation of IPSEs now emerging includes the flexible, comprehensive, modular IPSE, which can be used as a flexible system engineering workshop. Software Sciences' ECLIPSE is a notable example. This type of IPSE is the CASE world's equivalent of the flexible manufacturing systems (FMSs) used in modern CIM environments.

All these comprehensive CASE environments tend to follow one or another of two approaches—the prescriptive or the proscriptive:

- The prescriptive category is not tied to any one methodology and therefore allows for greater flexibility in the implementation of different formal methods within one suitable whole, for any particular project.

- The proscriptive approach means that the IPSE is tied to one particular methodology and is that much simpler to implement.

The trade-off in these two approaches is between simplicity and perhaps greater rigour on the one hand and greater tailorability and easier methods migration capability on the other.

Computerising the development life cycle leads to the identification of three levels of activity:

- Project Management, which handles the overall control of the development staff, their allocation to tasks and their cost-effectivness
- Tools that assist people in designated tasks, e.g., analysis, design, testing etc.
- Tools that automate certain tasks so that they no longer need to be allocated to people or have become part of another task that is now partially automated

Attempts to automate rather than to give computer assistance to the system development process have led to hightened expectations on the part of users and have given free rein to marketing hype. It is impossible fully to automate software design unless one defines the design in such a way that it is the design process of the system one wants to replace. Many advanced systems (such as those we have already described as application generators, 4GLs and the one system that calls itself a 5GL) claim to do away with the 'how' of the design process, putting it on the shoulders of the system to achieve this 'how' after it has been told the 'WHAT'. This declarative approach invokes an equal and opposite reaction, in that it actually pushes the design problem back a stage or up a level in the organisation.

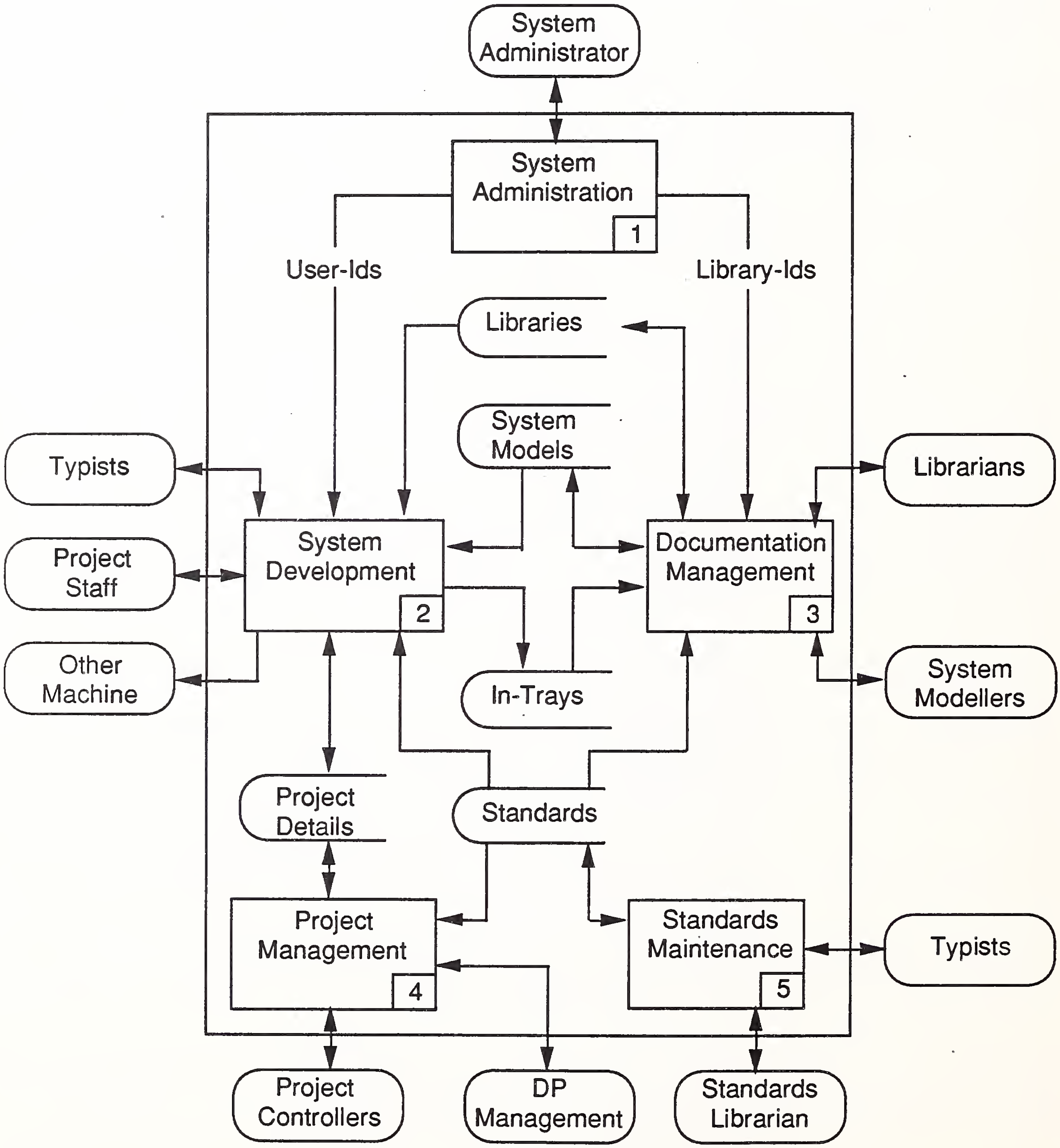
This is because all systems draw on existing basic data structures and need to know how existing data are held, who else uses it and how well these people understand the informational content. Since this is a non-trivial exercise in any but the most circumscribed set of the conditions, design, instead of disappearing, becomes a higher level task, and the organisation has to invent a new level of person to do it. For example, the system analyst inherits designs from the business analyst, who in turn inherits strategic systems details from the possibly mixed task force preparing the strategic plan.

Comprehensive environments (IPSEs) differ according to their approach to these multilevel tasks:

- The early or first-generation IPSE assisted the co-ordination of discrete tasks in a development team by automating and centralising information flow between members of the project team; an example of information flow is shown in Exhibit V-11; the diagram was drawn by one of the tools in the IPSE tool set;

EXHIBIT V-11

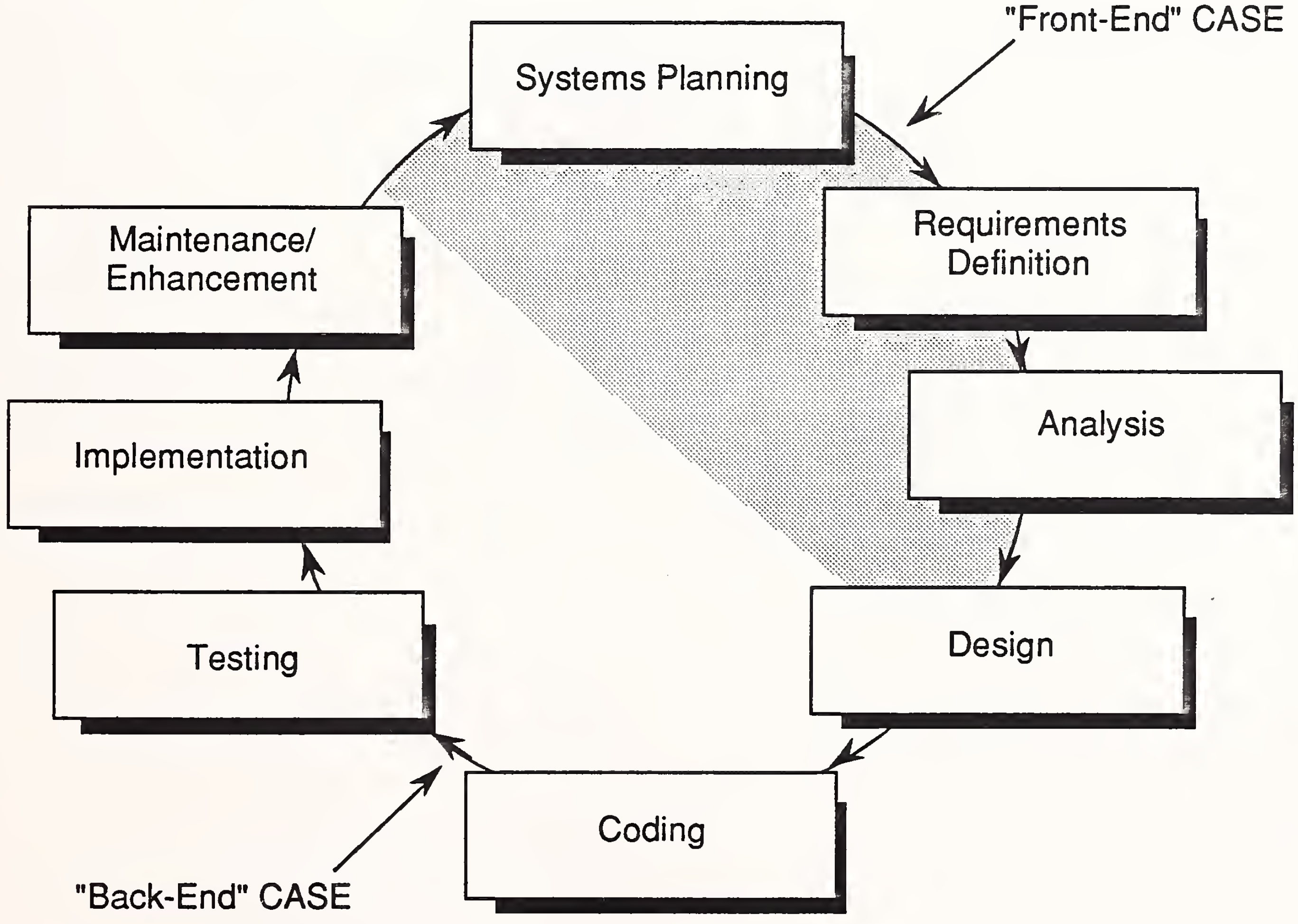
THE STRUCTURE OF THE BIS/IPSE



- I-CASE tool sets emphasise the linkage and possible automation of tasks in order to minimise misunderstanding, standardise usage and shorten system leadtimes; Exhibit V-12 shows the overall circular system life cycle previously described in unit terms, which the I-CASE vendors claim to support fully. Vendor claims are normally only partially fulfilled; in particular vendors have not been able to close the circle completely at the top by correcting back-end maintenance in one generation to front-end planning in the next. The reverse-engineering technology needed to achieve this goal has not yet been provided.

EXHIBIT V-12

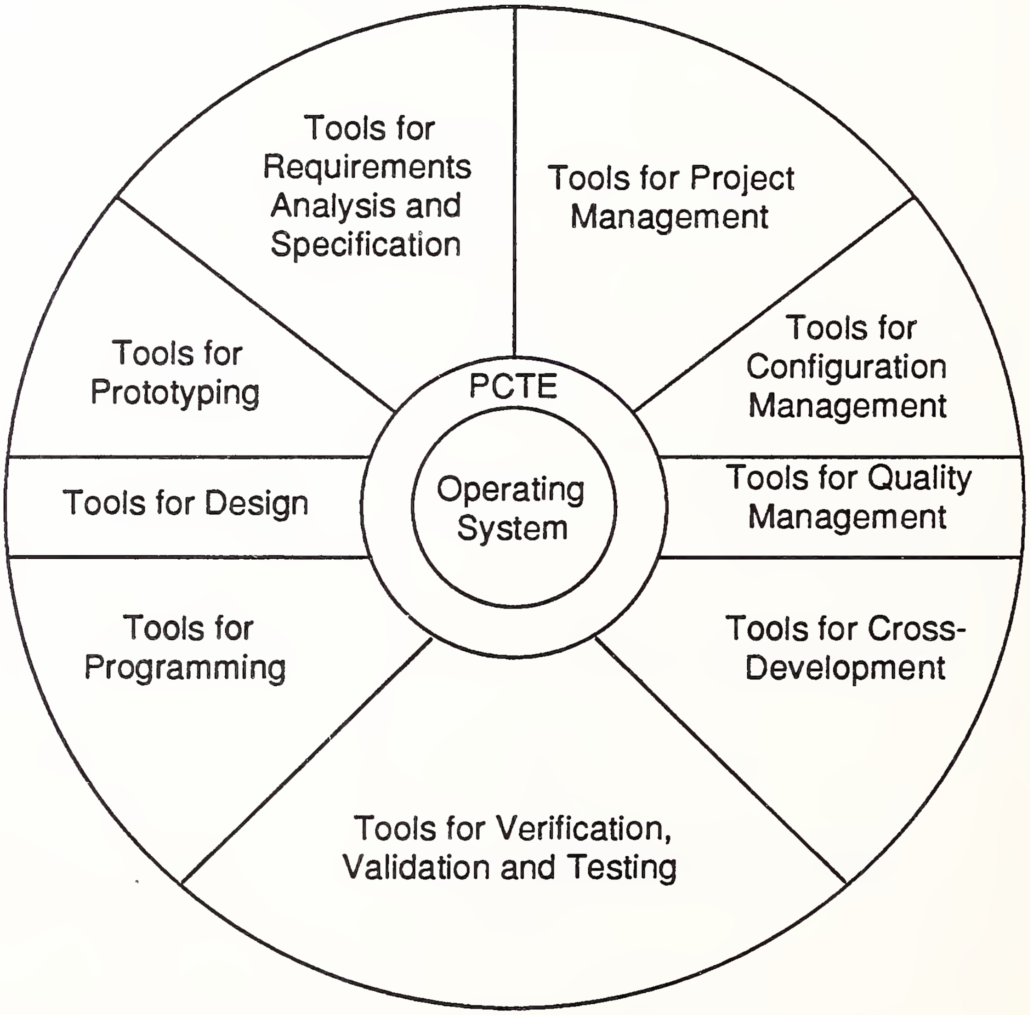
CASE SYSTEM DEVELOPMENT LIFE CYCLE
SHOWING FRONT AND BACK ENDS



- The third and next generation of IPSEs aims to allow for both possibilities, i.e., coordination and standardisation. The third generation does this by introducing the concept of the tailorable IPSE. Tailorable means a common interface approach to the linking of specialised tools to form tool kits tailored to the methodology and techniques chosen by the project management. This degree of freedom is the aim of the PCTE (Portable Common Tool Environment), an EC-sponsored standard developed as an Esprit project. This systems architecture is shown schematically in Exhibit V-13, in which PCTE is seen as the interfacing ring between the kernel operating system and the individual linkable tools in the life cycle circle.

EXHIBIT V-13

THE PLACE OF PCTE IN FUTURE IPSEs


Tools for Requirements Analysis and Specification
Tools for Project Management
Tools for Prototyping
PCTE
Tools for Configuration Management
Tools for Design
Operating System
Tools for Quality Management
Tools for Programming
Tools for Cross-Development
Tools for Verification, Validation and Testing

Examples of this latter approach are given by:

- ECLIPSE from Software Sciences, Farnborough, U.K.
- The new MAESTRO-IPSE from Softlab GmbH, Munich, West Germany
- SOFTORG from SES GmbH, Neubiberg, West Germany

All IPSEs have in common the characteristic of separating development from the run-time environment, with linkage between the two via the repository, encyclopaedia or project dictionary (whichever terminology is used by the individual vendor).

VI

Some User Perceptions

VI

Some User Perceptions

A
The Hardware Suppliers

All system manufacturers and hardware suppliers need to take an interest in the CASE market, mainly as providers of system software. The large computer suppliers are taking an increasing market share in both system and application software for the mini and mainframe markets. They therefore need to make use of the new CASE technology for production purposes, particularly in system software. Because of the high usage factor in the life cycle footprint of system software, the hardware suppliers are in most need of the application of strict engineering disciplines to this product line. However, suppliers are not necessarily the best people to advise on how CASE should be applied in other application areas where different life cycle footprints are found.

The IBM sector is moving at the top end towards reliance on IBM's DB2 relational database product. IBM is expected to announce shortly a repository based on DB2. This repository will be aimed at providing a format for all types of project data, from textual documentation through graphics to high-volume transaction data structures.

The IBM strategy for the next 10 years revolves around the SAA (Standard Application Architecture). This architecture allows for the free movement of systems across the whole range of IBM equipment. INPUT advises all vendors operating in the IBM field to take full note of the need for compatibility with SAA concepts.

In its recent announcement of the AS400 series, IBM stressed the inclusion of an application generator for use with this midrange system replacing the S38, at a time when disenchantment with the low-end 43XX Series has set in.

ICL has adopted an application generator with a front end that aims at the production/implementation level in a major CASE architectural framework.

Honeywell Bull addresses the CASE requirements on two levels:

- The information system engineering level (ISE), in which the detailed systems are developed

- The strategic planning level, in which the product SPA (Strategic Planning Approach) handles the best use of existing skills and the need for total management commitment

The Honeywell Bull ISE adopts a loosely coupled approach to the CASE sector, defined as including both tools and methodologies.

The SPA initiative calls for commitment from management at three levels:

- Top management
- Middle and operational management
- External consultant managers and other professional advisors

DEC is currently leaving most of the short-term CASE initiatives to its value-added resellers, putting on offer the RMS and RDB data systems as, respectively, a traditional file manager and a relational database. Longer-term, DEC's strategy revolves around in-depth use of expert systems.

The smaller size of systems encountered in the DEC DDP sector allows for greater use of 4GE and application generator products.

Nixdorf Computer has the ENCAPE project in hand as an internal tool to allow for the production of both system and application software. The strength of this system is gauged to be connected with its strong involvement in expert systems and AI technology. Nixdorf stresses the need to move software engineering in the same direction as automotive engineering.

Hewlett-Packard has adopted GEC software's GENOS IPSE for its internal use. Other minicomputer manufacturers are at the moment more concerned with the provision of departmental system hardware, increasingly based on the UNIX environment. Other minicomputer manufacturers are as yet little touched by the CASE sector and its issues.

B

Professional Services Vendors

The professional services companies have the greatest need for software development tools and methodologies. They also have the greatest number of specific software tools, developed for individual projects, in their existing tool kits. However, most of these tools are too specific to have a great amount of reusability.

On the plus side for their adoption of CASE, these vendors have many large projects over which to amortise the cost of buying or developing CASE systems.

On the negative side, there are, however, a multitude of different platforms over which these tools have to operate. In the past, building general tools has been deemed to provide insufficient performance and efficiency to favour the cost trade off. Most professional services companies distrust the large, generalised IPSEs; a minority of the larger vendors is very keen to see the I-CASE environments promoted.

As for methodologies, the policy generally is to use the contractor's own method unless the client stipulates otherwise.

- For example, CSC uses its own methodology called DSDM, which puts the emphasis on the provision of on-time delivery, and is strong on project management aids.

- CGS, Europe's largest professional services vendor, uses its own methodology called EXPERT, which stresses the overall importance of delivering quality in the system.

Most vendors use a packaged product for project management/control.

Analysis and design tools are reckoned by most systems houses to be of good quality and are used throughout the sector in the form of the analyst workbenches currently on the market.

Two of the most important IPSEs in the market-place have been provided by systems houses as turnkey systems:

- The MAESTRO product, produced by the West German company Softlab and sold by the Philips organisation, has been running on Philips P7000 mini equipment but is now being converted to using a standard UNIX platform and is at the same time undergoing a complete rewrite of its facilities in the framework of a loosely coupled modular mode.

- The MULTIPRO product from CGS, originally provided on DEC minicomputer equipment, is now sold via a PC platform by a US distributor, which is selling it under its own name on a worldwide basis.

Joint fundings by national and European Community research bodies are developing major comprehensive IPSE products based on the PCTE standard. Vendors involved in these projects are thus obtaining tools for their own use internally as well as being able to position themselves in the market as product providers.

Vendors express doubts about the transition of CASE tools from the design phase into the code generation phase because machine-generated faults are harder to trace. This area will need to wait until there are better tools to verify and validate the software by mathematical means.

Testing and debug tools have been on the market from the manufacturers and other suppliers for a considerable time. Integration of these tools into the correct comprehensive environment would aid the systems houses, the smaller ones particularly.

The best advertisement for the efficiency and quality of the systems being developed by these firms is for their client base to be able to see them using their own methodologies and their own tools in-house.

C
Real-Time Applications

The top-down approach to integrated CASE is less applicable in this application area, since modules have performance characteristics that must be discovered and tested in semi-live conditions before full specification of systems.

Hence the evolution of the loosely coupled IPSE approach embodied in one methodology and supporting a range of other methodologies and techniques. The IPSE for large, real-time systems must incorporate a compatible set of methodologies between the procurement stage, the systems analysis and design stage, the build stage and the overall project management strategy.

Many real-time systems also involve the development of specialist hardware that must be integrated with the software to be built. The evolution of hardware design tools is as yet not integrated with the development stream for software, as exhibited by the CASE market. The convergence of EDA (Electronic Design Automation), microprocessor development systems and CASE will not take place until the early 1990s.

Work is being pursued on special-purpose languages for the production of development tools to a common interface standard. The Hood System has been produced under contract from ESA by Software Sciences for use on the Columbus space project.

D
Commercial Users

The top-down approach, although theoretically unsound, is in practice favoured by INPUT for the large commercial DP user. Any methodology becomes diluted in its implementation in a large organisation. There is therefore the need for discipline and rigour of an imposed methodology to maintain the minimum of dilution, but this implementation is best made with loose coupling between the business strategic planning level (BSP) and the information systems (IS) level.

1. The Backlog Problem

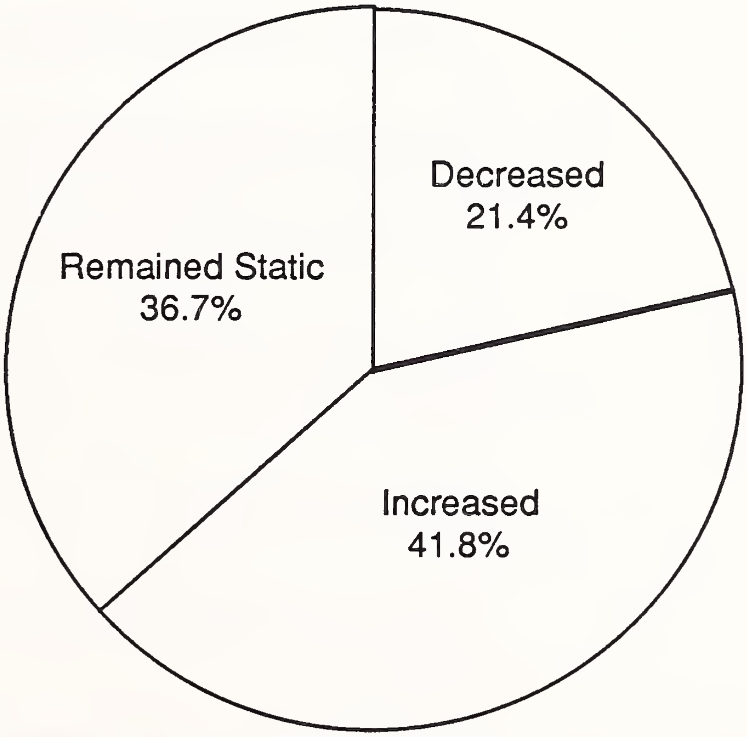
This section provides a summary and analysis of INPUT's user research in this topic area and highlights the trends and issues affecting the average commercial DP shop with regard to CASE developments.

As part of INPUT's ongoing research programme in Western Europe, an annual telephone survey is addressed to DP managements in medium and large companies (defined as having at least 500 employees). The questionnaire used is found in Appendix D. The sample for this analysis is 100 companies.

CASE is a subject to which the average system developer is so far not well attuned, but the presence of an ever-increasing applications backlog is well known and well documented. Exhibit VI -1 illustrates the way the backlog has changed over the course of the previous year:

EXHIBIT VI-1

APPLICATIONS BACKLOG— CHANGE BETWEEN 1987 & 1988



From 98 Responses Out of a Sample of 100

- Almost twice as many respondents reported an increase as reported a decrease, with just over a third maintaining the same level as previously (last measured as between 18 and 24 months in most cases).

- Seventy-nine percent did not decrease their backlogs during the year, down 7% from the 86% percent reported to be in this position one year ago.

In many cases, IS management has now come to see the backlog as an inevitable fact of life. Like the poor, it is always with them. In one situation, INPUT found a respondent who actually admitted that a controlled backlog was part of company policy to maintain high staff utilisation!

The most-often-cited reasons for the backlog are:

- Shortage of personnel and other resources

- Fast-changing technology

- Budget limits

- Legislative changes, demanding applications to undergo modification or even whole new suites to be developed

Fifteen percent of users reported that they were satisfied with their systems and the backlog situation. Many IS managements, however, perceive satisfaction as relating to hardware performance and capacity, as evidenced by the tone of their answers.

INPUT believes that the psychology of the backlog is intrinsic to the structure of the information industry, and is part of the dynamic tension between the major players in the market-place:

- The competition among the hardware suppliers creates an incentive to oversell expectations.

- The ability to go outside to have systems developed by software houses is a constant incentive to in-house teams to improve their productivity.

- The provision of end-user computing at both corporate and desk-top levels has only whetted the appetite of customers to get more-sophisticated systems, which their own time and resource constraints make it impossible to produce.

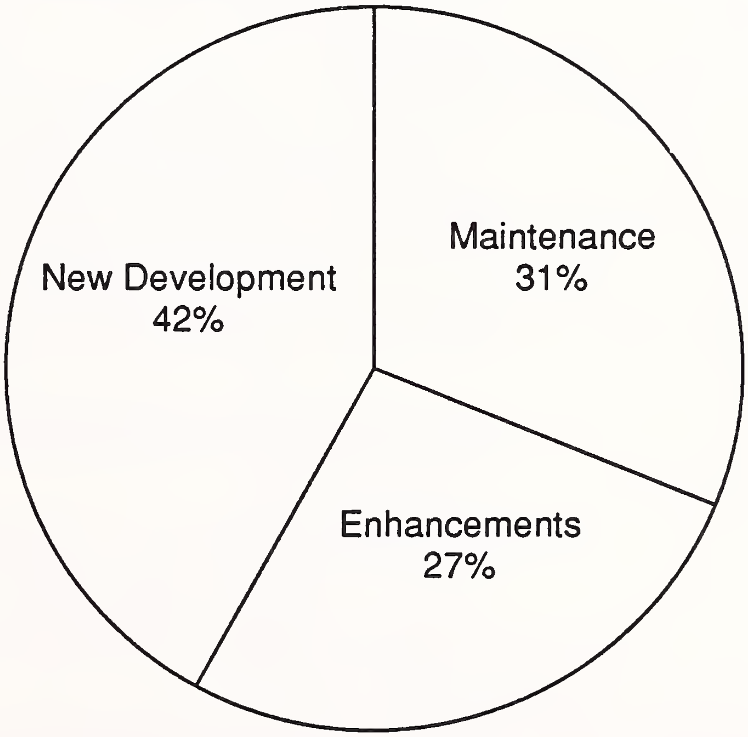
The maturation of CASE tools and systems could swing the pendulum back in favour of the in-house data centre solution, but in this next phase development must be planned to be built for DSD (distributed software development) environments, as seen in Exhibit V-2.

2. Allocation of Staff Time to Different Activities

Exhibit VI-2 gives the reported breakdown of development staff time on activities seen as devoted to extending the lives of existing systems or to replacing them with new ones. The often-quoted 60%-80% spent on maintenance is not supported by this research. Even if enhancement is lumped with maintenance to make the figures look worse, we are only looking at 58%— near enough to 60% to rouse the suspicion that many surveys stressing the horrendous nature of the software maintenance load do not make a proper distinction between the two.

EXHIBIT VI-2

APPLICATIONS DEVELOPMENT STAFF TIME SPENT ON MAIN ACTIVITIES



Out of a Sample of 100

- The cost-effectiveness of maintaining/enhancing systems has usually been higher than replacing them, although this gap will close, as and when CASE systems provide real choices in this area. Replacement will require the provision of reverse-engineering technology applied to systems built to known and proven methods.

3. Involvement with CASE Technology

Twenty-three percent of users reported involvement with CASE systems, defined as products to support all phases of the development life cycle. Among these, the commonest primary reasons given for purchase (in order of decreasing importance) were:

- Improved speed of development/productivity for 57% of respondents
- Improved quality using a standard approach for 26%
- Flexibility and ease of use for 13%

The first two reasons cited here are given in the reverse order to the two most important benefits promoted by INPUT's vendor sample when selling to prospects: Vendors without exception stress quality before productivity.

INPUT interprets this reversal as indicating the day-to-day pressure on IS managers, which could well in the long term militate against the proper adoption of tools and methodologies to produce better systems. Keeping the end-user at bay could mean that the next generation of systems should be thought of as throw-away plastic systems, rather than the robust, well-engineered systems dreamed of by the developers of CASE.

Major difficulties encountered in evaluating CASE were reported (in decreasing order of importance):

- Adapting to the user's own needs—9% of CASE users
- Performance problems in high-volume transaction systems—9%
- Time and money invested—9%
- Deciding objectives, choosing a tool kit and training—5% each

Twenty-seven percent of respondents reported no major difficulty, and 14% were still evaluating or just starting to.

Of the 15% of the sample who already had some experience of using CASE:

- Four reported that their implementation had proceeded without any major problems.

- Eight had had difficulty because of needing training or familiarisation.

- Two had had to work hard to raise interest on the part of their development staffs.

- Interfacing problems had claimed one victim.

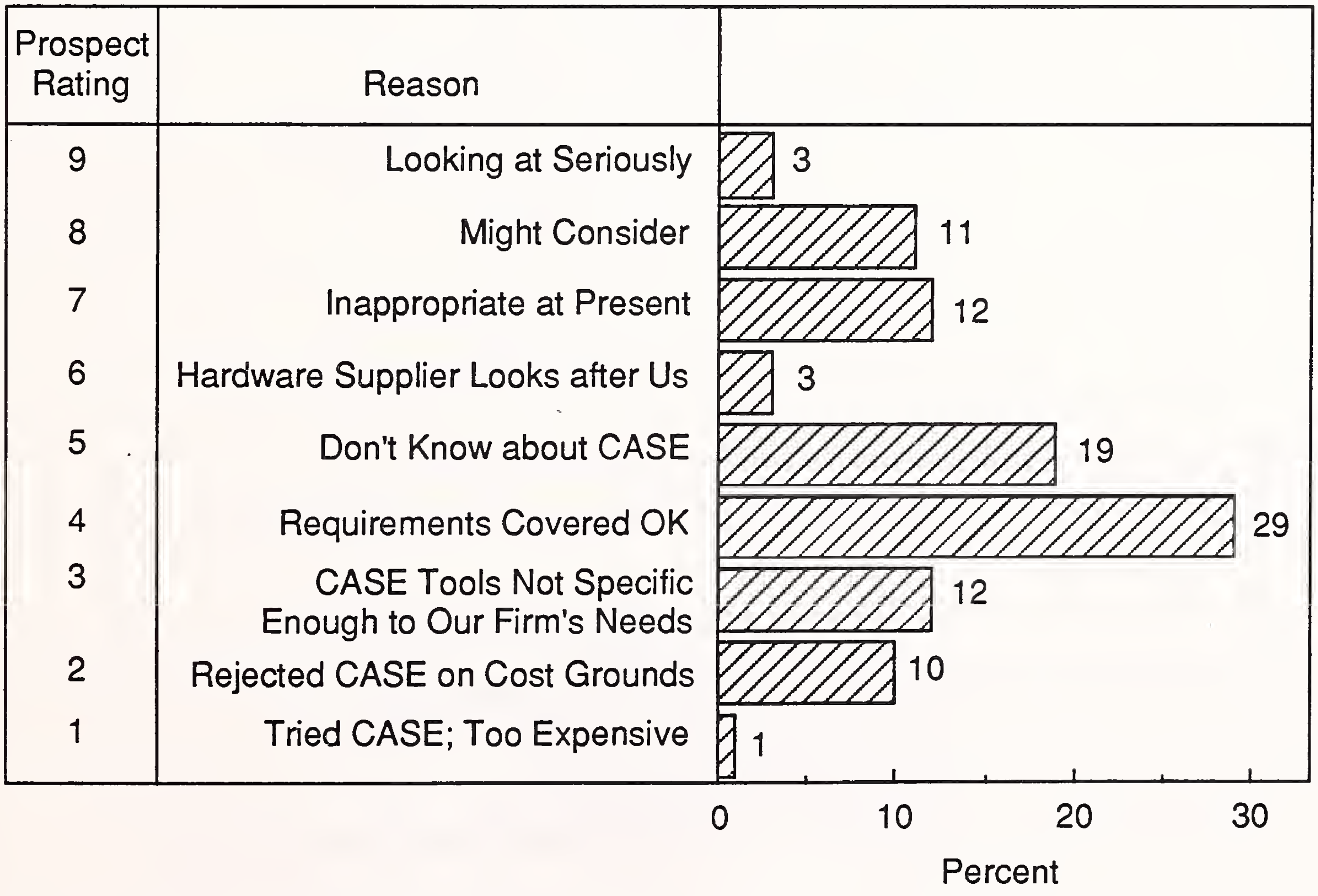
4. Rejection of CASE

Seventy-five percent of the sample had not adopted CASE.

The percentage breakdown of that portion of the sample is shown in Exhibit VI-3 as a bar chart spectrum sorted by ascending order of their value as current prospects, as estimated by INPUT. The distribution clearly peaks in the middle of the chart, with 29% satisfied with their current situation and 19% of nonusers too ignorant of the possibilities even to make a judgement. This order, of course, could easily change as technology and business requirements bring about changes to DP management's priorities. Today's doubters can be tomorrow's champions.

EXHIBIT VI-3

STATED REASONS FOR NOT ADOPTING CASE
(Out of 75 Responses)



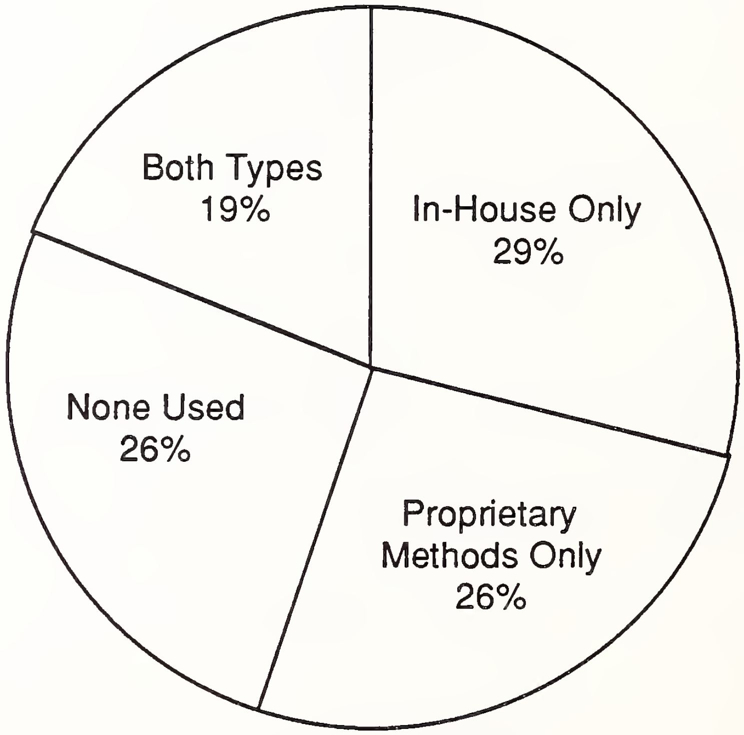
5. Structured Methods

Almost three-quarters of the sample claimed to be using structured methodologies. As shown in Exhibit VI-4:

- In-house methods were most favoured (by 29%).
- Proprietary methods followed with 26%.
- Nineteen percent claimed to use both types.

EXHIBIT VI-4

USE OF STRUCTURED METHODOLOGIES



Out of a Sample of 100

Among the 26% who did not use any structured methods, the following reasons were given:

- Now considering using them (1 respondent)
- Too small to need them (3)
- Suppliers or consultants look after us (6)
- We don't know about them (4)
- Requirements covered OK (5)
- Not suitable to our needs (4)
- Impossible to use them for staff or organisational reasons (3)

Eighty percent of these respondents were also non-CASE users.

6. Summary

Users were finally asked to state their priorities in improving the software development process. Exhibit VI-5 shows their responses.

- The top three factors are all qualitative imponderables relating to good teamwork between the DP professionals and their internal clients. Project management skills are more important than either tools or methodologies (in that order).

EXHIBIT VI-5

FACTORS IN IMPROVING THE SOFTWARE DEVELOPMENT PROCESS

Possible Actions	Their Ratings
Increasing Commitment to Quality	8.13
Increased End-User Involvement	7.85
Increased Awareness of Business Problems	7.82
Training—Project Management Skills	7.18
Training—Tools	6.99
Training—Methodology	6.96
Change in Organisational Style & Culture	5.68

INPUT concludes that:

- CASE tools as a sector has only penetrated a quarter of the large and medium-sized organisations in Europe.

- Only a quarter of those who have used CASE tools have done so without any major problems.

- Unsuitability and cost were the two main reasons for rejection of CASE by those who had actively investigated the possibility of using it.

- Satisfaction with the status quo and ignorance were the two main reasons for nonadoption by those who had not formally evaluated CASE.

- Approximately 10% of nonevaluators are coming close to looking at or feeling a requirement for CASE at the present time.

The market for methodologies is much more saturated:

- Seventy-five percent of these same users have some experience using methodologies, although INPUT has not recorded their implementation problems.

- A variety of reasons is given for not adopting a formal methodology; no particular one stands out.

- CASE users are very much more likely to be users of a structured methodology than not.

- There is in both areas a great need for education, training and general hand-holding, not least because of the degree of ignorance professed by IS managers in the face of the fast-changing nature of information systems.

VII

Technology Streams

VII Technology Streams

Several technology streams are converging upon the CASE marketplace as well as pursuing their own paths of development in or towards application areas specifically suited to CASE features:

- Knowledge-based systems (KBS) technology
- Embedding of software intelligence in firmware
- Reusability engineering
- Networking software requirements
- Object-oriented programming (OOP) techniques

A Knowledge-Based Systems

This term is now preferred among systems suppliers who used to be known as AI vendors. These vendors have in certain subsegments suffered badly from the reaction to the marketing hype of previous years and prefer not to use a terminology reminiscent of that era. Nevertheless, substantial growth is being experienced by suppliers of cost-effective solutions in distinct sectors. Two of those sectors are:

- Expert systems
- Natural language systems

1. Expert Systems (ES)

We have seen how this technology has become embedded in a range of system development tools, including 4GLs and 4GEs that use rule-based techniques to assist in the code, database linkage and JCL generation functions. This results in more comprehensive and efficient application generators reaching the market and either taking over from earlier 4GL products or making for powerful program generators that can be interfaced to the back end of front-end CASE tools.

INPUT has also seen front-end CASE workbenches being used by development staff working on Expert Systems—up to the stage of the data flow diagram. This is a case of symbiotic growth of the KBS and CASE sectors, with of course the professional developer playing a fundamental role in the design/production loop.

In its own right the Expert Systems (ES) sector has a multimillion-dollar market worldwide in 1988.

Most of the major hardware suppliers are engaged in large-scale R&D efforts to make use of the power of KBS:

- DEC's Database Adviser aims to assist in the selection of the correct database for supporting an application; the chosen role is as a front-end technical support tool in an increasingly multidatabase environment.

- Configuration of elements in a large piece of software, the kind of problem encountered in big military software projects, is a problem being addressed by UNISYS.

- Code generation for PL1 is being assisted by similar techniques in one of IBM's programs.

A key commercial characteristic of the market, stemming from the nature of ES technology, is the separation of the development system from the delivery system, as illustrated in Exhibit VII-1. In this respect, ES points the way to all future CASE environments, in which source or problem/ design considerations need to be kept away from target or production considerations.

Falling hardware prices and the needs of good professional practice will make ES an industry standard in the 1990s:

- Front-end analysis and design up to the testable prototype stage must take place on a separate physical platform from the final target platform in all but the simplest systems.

- Engineering practice dictates that design engineers have their sights firmly fixed on the specific product on the drawing board while the production engineers look after the integration of the product into the overall production line.

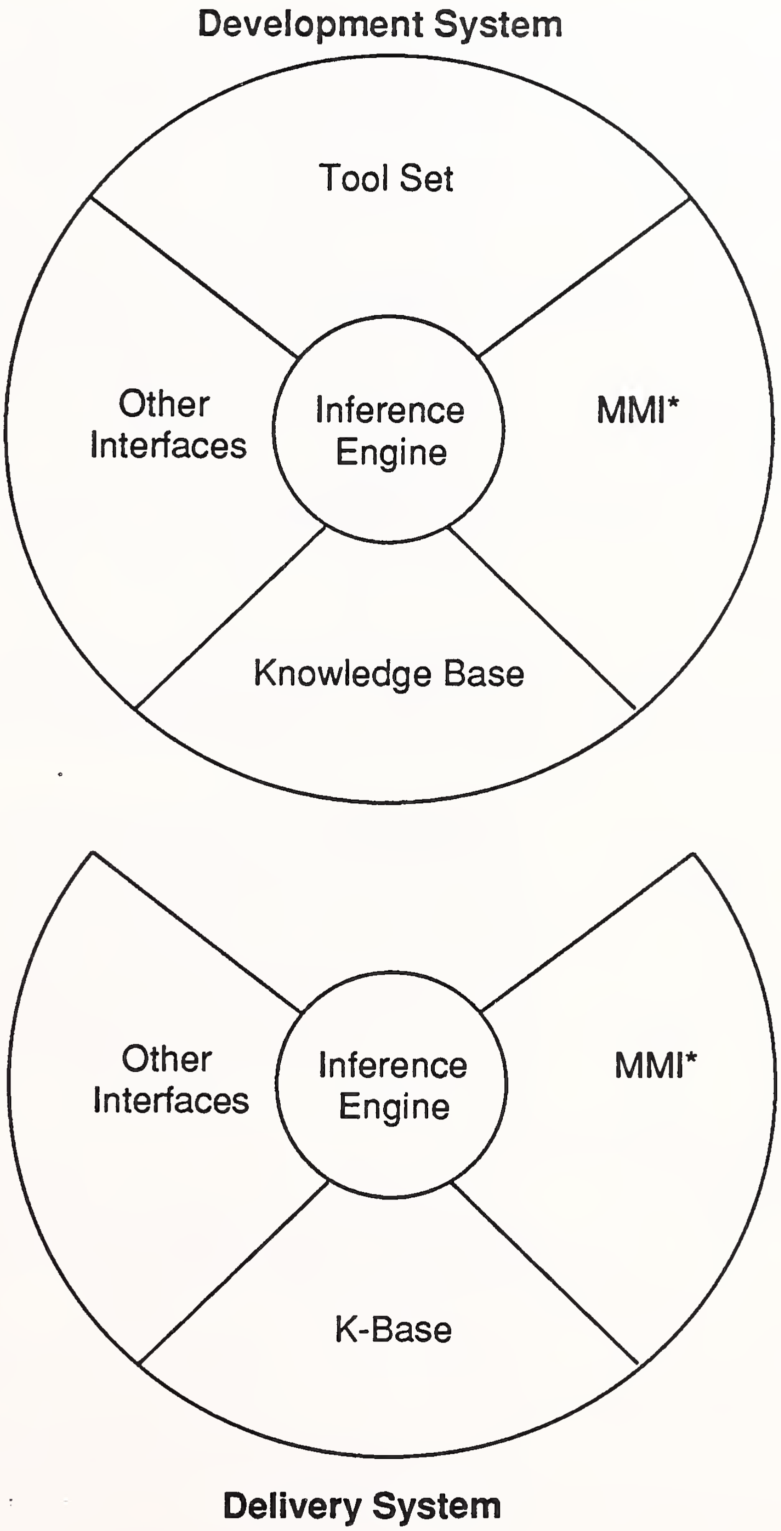
- Increased use of server architectures will enable this standard.

2. Other KBS Developments

The use of advanced prototyping tools requires the knowledge and training of systems engineers with experience of these techniques. Other systems allow the intelligent technical professional to use them in day-to-day work:

EXHIBIT VII-1

EXPERT SYSTEM ARCHITECTURE



* Man-machine interface

- INTELLISCOPE is a set of software routines from Intellicorp that helps DSS users access information in complex knowledge and databases without recourse to the enigmas of SQL.

- REFINE from Reasoning Systems can represent the specifications of a system in a high-level form and allow the developer to browse through the system and see different views.

- DATATALKER from NLI (Natural Language Inc.) is geared to letting users access information in a database using simple English. DATATALKER is used after a module called Connector is used to turn the informational content of the database into the correct representation required by the Knowledge Base structure in use—again a job for the professional, unless the database is private or simple.

- GENERIS from DSL (Deductive Systems Ltd.) is a tool formed from an intelligent database. GENERIS allows the input, updating and interrogation of knowledge structured in a knowledge base.

In all cases the trade-off between self-use and using a professional should be made on a common-sense commercial basis. Vendors must assist users by adopting sensitive attitudes to this dilemma within their marketing programs.

The Japanese Fifth-Generation project, which did so much to stimulate U.S. and European research in advanced, KBS-based systems engineering, has been dubbed a failure by some in the West, but its project manager has recently defended his progress, saying that detractors have misconstrued the objectives. The aim, he says, was always more limited than publicity implied; the objective was to build and demonstrate a logical inferencing machine, and this goal is on target.

B

Software, Firmware and Hardware

The concepts behind software engineering have their origin in the more mature disciplines associated with hardware design. Many analogies can be drawn between the two areas and are so drawn, usually to the detriment of the younger discipline.

Vendors have to understand the similarities and the distinctions:

- Pure software systems are not normally tested until they are built—the prototyping bind.

- Pure hardware systems are normally tested in real-life conditions by building a model and testing that.

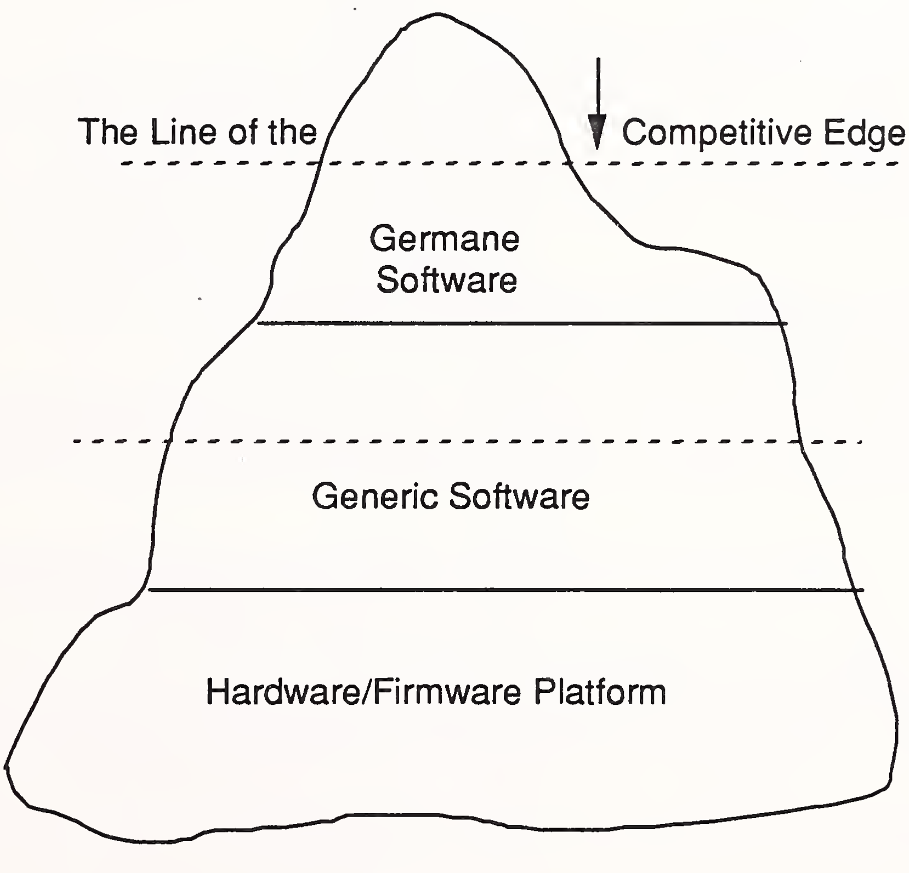
- As software matures, each side borrows techniques from the other—hardware uses software simulation to cut prototyping costs, and software now borrows hardware's disciplines, including the building of prototypes.

Hybrid hardware/software systems present the most difficult case because the two sets of disciplines must be synchronised in a teamwork environment. Increasingly the software provides the leading or competitive edge, as illustrated in Exhibit VII-2, which shows the steady productisation of software into firmware/hardware in the market. The situation is analogous to an iceberg, where:

- Generic software has to be provided at market rates.
- Germane software is effectively priceless.

EXHIBIT VII-2

THE SINKING ICEBERG



Selling to users means knowing where the line between the two can be drawn at any time. Software and system vendors know that the productising of software can mean that its hardware implementation becomes a possibility, whereas near the top of the iceberg as soon as germane software becomes common knowledge, it can become generically applicable to that particular industry sector.

Some vendors of CASE tools come to the market from a hardware design background:

- Mentor Graphics, the leader in the EDA (Electronic Design Automation) graphics field, has acquired the CASE products division of Tektronix, although Tektronix has kept the microprocessor development systems side of its business, where it was already a market leader in the embedded systems market.

Some systems houses have developed tool sets for the hybrid hardware/software embedded systems area, and are now marketing these as commercial products.

- Marconi Software Systems is offering TYPHOON, a host/target cross-product development tool kit that enables a high degree of independence between the hardware and the software development paths by allowing special-purpose hardware to be simulated in software prior to its construction, thus familiarising software engineers with the parameters of the hardware for their own testing purposes and taking them off a delayed critical path.

Another software company is involved in the marketing of a complete development environment for the production of application-specific integrated circuits (ASICs):

- ELLA (ELectronic Design LAnguage) was developed originally at the RSRE (Royal Signals and Radar Establishment) and is now marketed by Praxis Ltd. of Bath.

- It is now being applied to verify the behaviour of the VIPER 32-bit microprocessor developed at RSRE. VIPER is claimed to be the world's first formally specified and verified 32-bit processor; it has been designed for safety-critical applications.

IBM has applied rigorous mathematical techniques to the analysis and design of some of its Federal Systems Division software in order to reduce the as-built software error rates and predict the MTBFs. One of the techniques involved testing with random data an integrated, modular system previously untested (i.e., in which the modules had at that stage not undergone any traditional unit or integration testing). This testing enabled the team to observe the appearance of bugs and measure their frequency of occurrence. From past bugs it was possible to predict future failure rates.

Research in optically interfaced systems/computers, coupled with research into simulation in software/firmware of the processes of the human brain (neural computing), has been pursued in the U.S.A., Japan and Europe for some years. The first commercially available PC-based software products have been purchased by blue-chip companies for development work in this area. In June 1988 Paris hosted Europe's first Neural Networking conference. This technology stream will have far-reaching repercussions in the mid-1990s for all products with embedded KBS technology.

C

Reverse Engineering, Re-engineering and Reuse of Code

The top-down approach to system engineering (championed with great fervour by James Martin and his disciples) needs to be complemented by tools that approach from the bottom up and

- Allow existing systems and modules of systems to be used as the building blocks or elements of new developments, whether for enhancement of old systems or for complete redesigns

- Stop the reinvention of the wheel every time changes in technology or business requirements dictate

Reverse engineering is a technique that enables IS professionals to extract business rules from old applications and use them, as appropriate, for the refurbishment or maintenance of existing applications, or even as part of the analysis of a new system rewrite—the equivalent of the trip round the organisation to find out what existing practice is, as taught in classical systems analysis.

Current reverse-engineering technology operates at three levels:

- It can be used to examine and report on the structure of existing programs, usually COBOL programs, so that the designer gets an idea of how well structured a program is, thus gaining the knowledge to assess its future maintenance cost or plan its continued maintenance.

- It can be used to put a good structure into a badly structured COBOL program, thus improving maintainability.

- It can be used to extract the physical and logical models of existing production databases by examining their contents at the data dictionary level.

Some companies active in this field at the moment are:

- Peat Marwick McLintock, the U.K. arm of the international accounting and consultancy practice, markets two products that use ES techniques.
  - PATHVU analyses and reports on the structure of IBM COBOL programs.
  - RETROFIT cleans the structure automatically.
- Language Technology Inc. (LTI) of Salem, Mass. in the U.S.A. markets two products, RECODER and INSPECTOR, that carry out similar functions and also draw on ES technology.
- Delta Software Technologie AG of Schwerzenbach, Switzerland markets Delta/SMP (Structured Maintenance Package), a tool set for the maintenance of COBOL programs.
- Keith London Associates of Hemel Hempstead, U.K. provides a reverse-engineering capability in a tool kit under the family name of METASYS, which allows for front-end design work and extraction of reports from existing data dictionary and encyclopaedia systems.

Exhibit VII-3 shows in schematic form the feedback loop principle as applied to the system life cycle: reverse-engineering allows the salmon to jump backwards up the waterfall. Shown here connecting code with design, the feedback loop principle could be applied between any two stages.

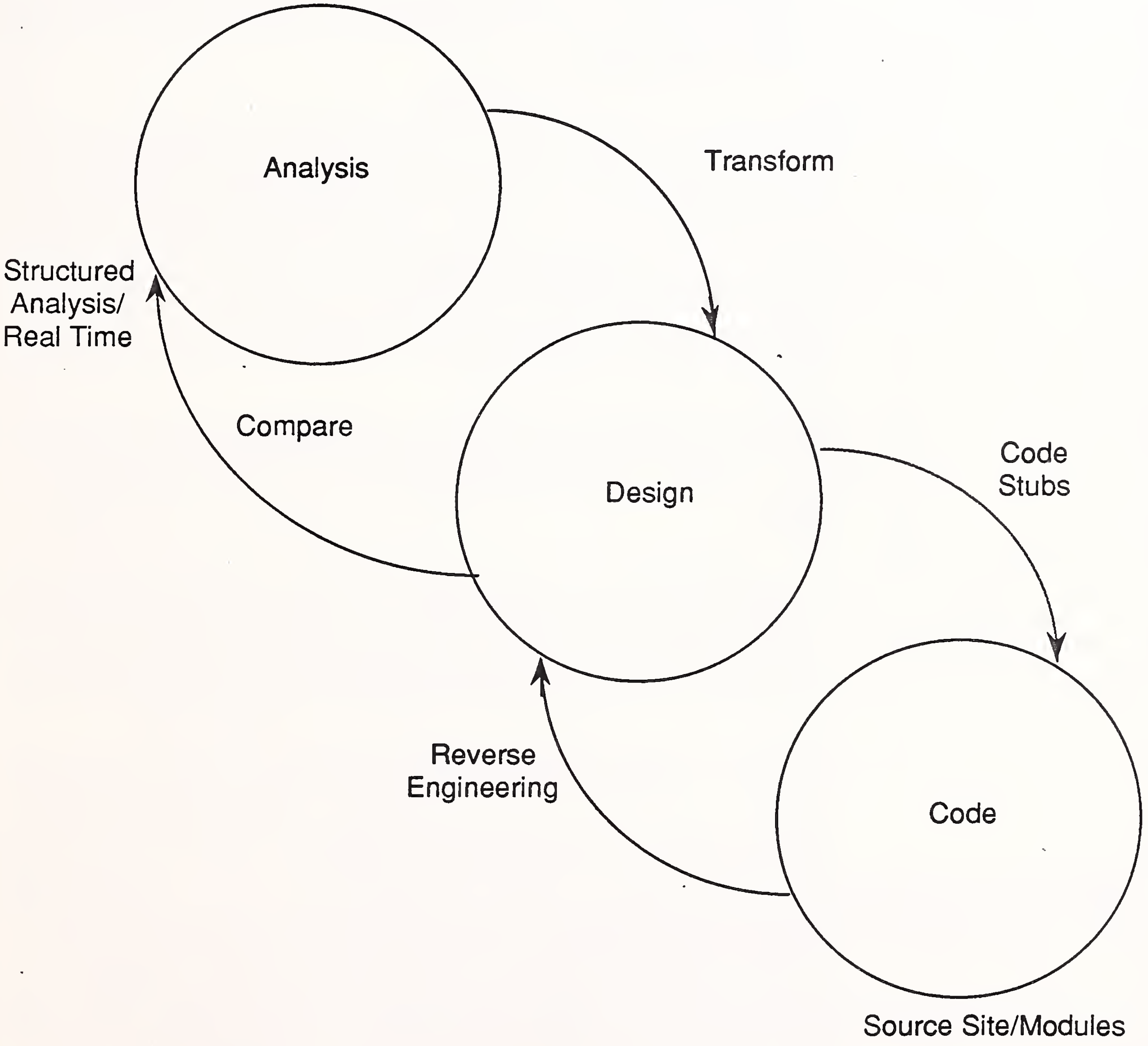
Present-day tools are designed for interactive use by the programmer/designer. They have been initially used to improve the maintenance function by producing better documentation and a cleaner structure.

- This process has been called software renewal.
- Aimed principally at the over-70-billion lines of code estimated to be in use in IBM COBOL production programs, the technique offers a strategy for rebuilding the urban sprawl and decaying tenements represented by the mass of undocumented programs (with unknown machine code patches and all the rest of the inelegancies to which these products are subject) on which many organisations increasingly rely for their survival. Because of their relatively high price and the unglamorous nature of their functions (who wants to be associated with a career in maintenance—the systems equivalent of the road sweeper?), these products have not received the attention they warrant.

INPUT believes maintenance, like programming, must be given its true importance in the scheme of things.

EXHIBIT VII-3

THE REVERSE-ENGINEERING CONCEPT



Another company (whose president is Charles Bachman, one of the pioneers of database management systems) takes this technology a stage further and builds it into the whole process of the continuous system life cycle—systems continually come up for renewal after suffering gradual degradation.

- Bachman Information Systems of Cambridge, Mass. markets two products, Data Analyst and Database Administrator, that can be used to extract logical and physical models from databases to improve existing designs and propose new ones.

Bachman's view of the complete life cycle process looks like the schematic shown in Exhibit VII-4. Reverse engineering climbs up the system decomposition stack, while forward engineering allows for the more usual top-down decomposition associated with front-end CASE tools and methodologies. Because of the possibility of having to reverse-engineer system modules built on previous or outdated methods, this approach implies a flexible methodological framework. It cannot work rigidly within one methodology unless all previous sytems using other methodologies have been replaced.

In smaller organisations not all the levels may exist or they may have been collapsed into each other by the use of a 4GL that bridges the implementation and specification partition; nevertheless the principle applies.

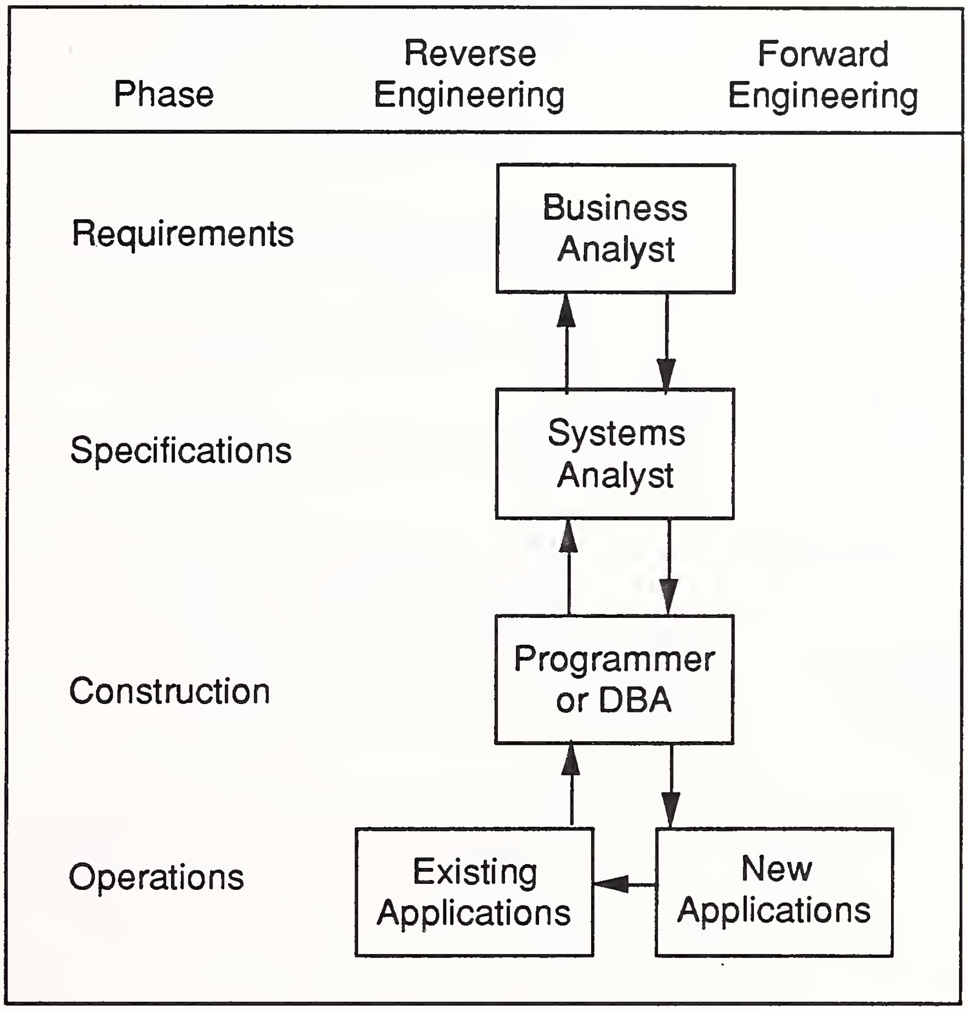
- If applied in the 4GL case, it would point out the need for a technology for formal prototype testing. To prevent the threatened proliferation of incompatible quick-and-dirty systems produced by the present generation of powerful 4GLs, suppliers should provide methods, tools, training and consultancy for this market gap (see Chapter VIII).

The term *re-engineering* takes in the complete life cycle as envisaged by Bachman. At the lowest level it involves the reuse of code and system kernels, as practised by professional services vendors, OEMs and package developers—in less formal ways, perhaps.

Another form of reuse is now being offered in conjunction with some 4GEs of recent introduction; this form is called template programming, in which the software producer supplies the core of a reusable 4GL-generated application system and markets this core through VARs, which can, using the same 4GL template, easily add value in the form of enhanced code or more industry-specific code. Template programming is an attempt to market tailorable or tailored products at package prices.

EXHIBIT VII-4

THE BACHMAN VIEW OF THE RE-ENGINEERING CYCLE



D Networking

The majority of data networking systems implemented in the late 1980s consist of terminals linking to individual host mainframes, each of which is hosting a range of databases proprietary to that machine site.

- The present situation is one host to one or more databases.

- The inverse situation of one database spread across one or more sites, known as a distributed database (DDB), does not yet command any support.

This situation is expected to change by the 1990s as interconnection of WANs and LANs leads to more-complex and more-demanding network applications. This change will have repercussions for the implementation of the DSD (Distributed Systems Development) environment described in Exhibit V-2.

- Interconnection between developers in different parts of an organisation will be more controllable from the central standpoint of the systems manager because there should be no need for duplication of data on different databases and therefore less likelihood of proliferation of private databases.

- More-sophisticated tools will need to be available to systems and operations staff to keep track of data and plan resources for the future.

In the late 1980s DDB technology is still in its infancy. Issues arising are:

- How to maintain the integrity of the central encyclopaedia or repository

- Standards for data interchange between communicating development processes. At present, EDIF and IRDS are being considered as drafts by ANSI and ISO, respectively, while the CASE and database vendors are operating on too limited a horizon see much early benefit in getting involved—for the most part, that is.

- The existence of proven DDB Management software. This is just emerging from the database product houses, with INGRES/Star an early example. The technical problem of efficient and secure polyphase commitment in the updating process has to be overcome.

Exhibit VII-5 shows the three-tier architecture of the INGRES/Star product—with host, DDB manager and back-end retrieval engines as the three levels.

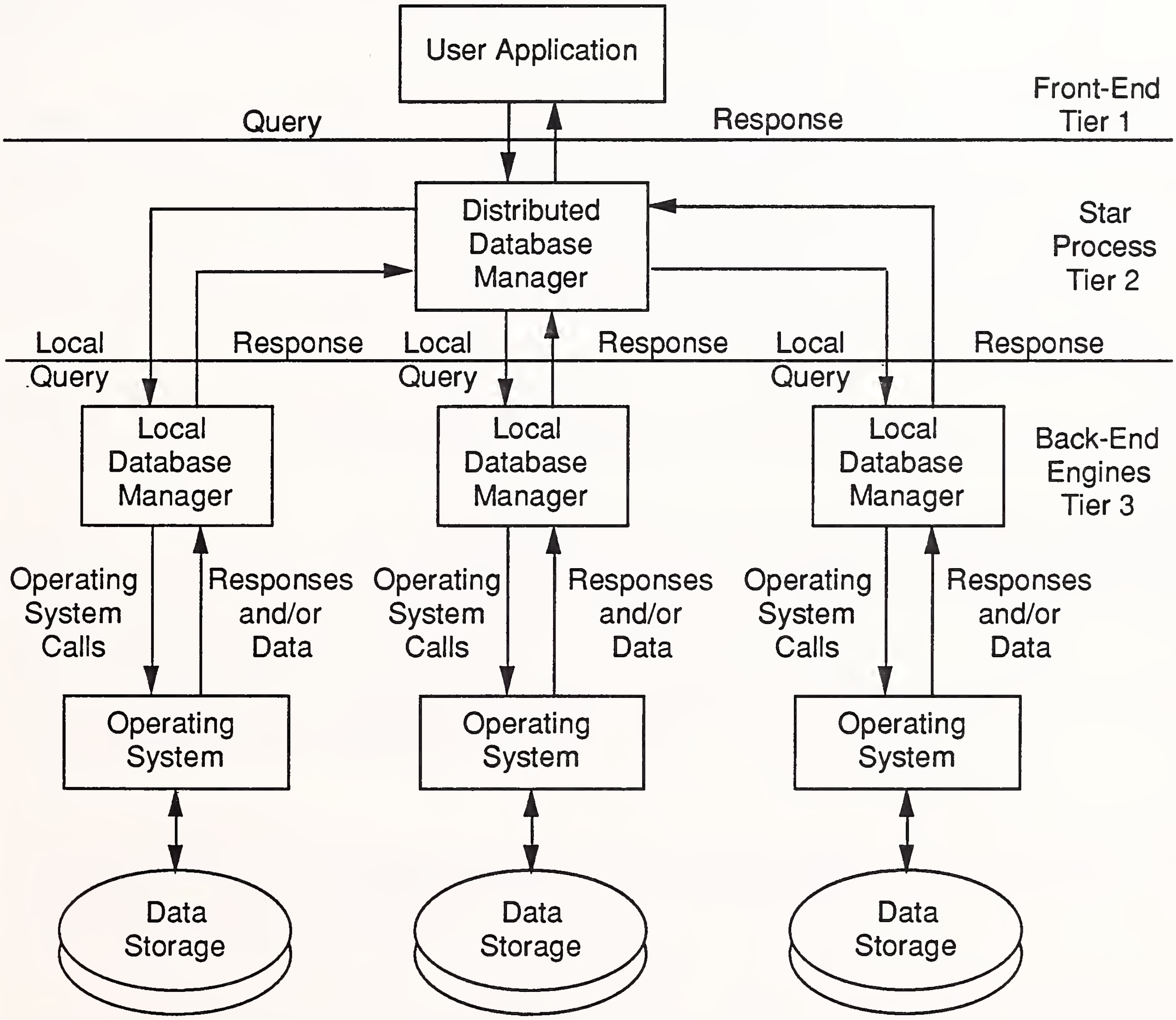
Central to the impact of DDB on systems engineering is the widespread adoption of a two-tier physical development environment in which development on PC, PS2 or UNIX workstations is kept separate from the mainframe/mini-based production environment. This report has alluded to the necessity for the separation to evolve quickly as a standard, but at present the waters are being muddied by the latest batch of 4GL products that cut across the boundary between development and production.

Vendors active in this area are:

- IBM, whose System Repository is expected to be announced shortly. IBM has to balance the requirements of SAA and DB2, which are strategic initiatives expected to cover the next 10 years, against emerging networking standards requirements.

EXHIBIT VII-5

THE 3-TIER STRUCTURE OF THE INGRES/STAR DISTRIBUTED DATABASE MANAGER (DDBM)



- Sun Microsystems, the leader in the 32-bit workstation field, on whose platform are expected to run much of the more sophisticated CASE systems required for military and real-time industrial systems developments. Sun has announced NSE (Network Software Environment) as an integrated generic development platform that was originally developed for internal use, but is now being targetted to aerospace companies and government and defence contractors. NSE has an interface for the integration of future CASE products and supports various development objects—source code files, design documents, drawings, test data and drivers etc.

- Software AG, whose ISA (Integrated Software Architecture) mirrors IBM's own SAA, is providing tools for systems and operations staff to use 4GL products to cooperate through its PREDICT data dictionary.

- Henley Business Software Ltd., a U.K. startup, is marketing Gupta Technologies' SQLWindows Design package aimed at the co-operative processing environment of linked PCs, network servers and mainframes.

The majority of vendors interviewed by INPUT for this study claimed to be adopting an open architecture approach, with UNIX and SQL the most-often-quoted planks in vendor strategies.

E

The Object-Oriented Solution

Object-oriented programming (OOP) was first conceived in the 1960s in Norway. It is a style of programming in which the real world is viewed as a set of objects whose behaviour has to studied and modelled. The problem can then be modelled in terms of objects that interact in the real world to produce the problem. The statement of the problem starts with descriptions of these objects, in terms of their names, attributes and interactions. Object-oriented languages (OOLs) exist to facilitate this process.

The development of OOLs went from early experiments with simulation languages for complex processes, through convergence with symbolic processing in the original LISP machines for AI work, to adoption by CCITT in the form of a language for programming telecommunications applications (SDL).

The wider interest in the use of OOLs has been engendered by the need to provide more enhanceability in real-time systems. Traditional programming has concentrated on designing round the processes of any real-time system; these processes have been found to be susceptible to change. The objects taking part in the processes change more slowly; hence basing a design round them is thought to be capable of producing more easily enhanced systems.

Three important characteristics are stressed by the proponents of OOP:

- OOP systems are modular
- They are extensible and modifiable
- They possess a feature called inheritance

Inheritance is a way of expressing commonality between the attributes of different objects to allow objects that belong to a subclass to inherit attributes from their superiors in the hierarchy.

Together these characteristics lead to a high degree of reusability of the object modules in an OOP system. Since reusability is one of the long-term keys to completing the system life cycle loop and enabling re-engineering to happen, one can understand the gleam in the eye of CASE tool set and IPSE developers when this technique is made standard. The way is opened for the building of standard components of software, the lack of which has rendered system engineering similar to hardware engineering only in name. What hardware engineer would think of embarking on a design without access to a handbook of standard chips, ICs and designs?

Work on this technology has a long history. Present-day developments of interest are:

- EC-funded research projects under the ESPRIT, EUREKA and RACE titles all incorporate a component of this technology that is familiar to the major European systems houses working in defence and other large-contract areas.

- Scandinavian research institutes are producing CASE tool sets for later commercialisation on the software market.

- C++ is an example of a language that supports OOP.

- Many of the advanced KBS tool sets (such as KEE, SMALLTALK and LOOPS, to name a few) are capable of supporting this technique.

Finally, a word of warning:

- OOP is expected to become widespread in large safety- and mission-critical applications.

- One of the features of the OOP technique is that it makes no distinction between the logical and physical representations of an object. Its more natural way of looking at the world removes the need to make this distinction.

- A speaker at a recent European conference on safety in systems, when asked to review the apparent slide of the industrial world towards disasters of greater and greater magnitude, said that: 'Disasters happen in the physical world and not in the logical world of the designer's mind'. Managers commissioning mission-critical systems therefore need to know the trade-offs between: a) ease of development (aided by OOP technology), which is essentially a commercial consideration; and b) hazards to the environment and the system's users.

VIII

Short-Term Opportunities

VIII

Short-Term Opportunities

A Market Trends

It is a long time since structured methodologies and CASE tools emerged from the R&D laboratories and into general use. No longer are they even the exclusive province of large companies.

The last two years have seen a fast uptake of front-end (Upper Case) tools and a rash of new products in a number of areas:

- Analytical aids
- Application generators
- I-CASE systems

The first disappointments have occurred:

- Benefits have not come through as quickly as users might have wished.
- It has been difficult to integrate the new tools with the existing methods
- The cultural change required in moving wholesale to CASE has in some cases proved unacceptable to in-house systems staff, or at best indigestible.

As a result of the use of the new standalone CASE tools, one realises the need to act within the framework of a methodology.

Meanwhile methodologies have been developing in parallel with the CASE tools, but here again scepticism on the part of users has been encountered. Some methodologies have been found to have drawbacks similar to the tools:

- They are too costly.

- They only cover part of the development life cycle.
- The early ones were designed as paper-based systems and do not convert well to working with automated aids.

In summary we can say that the present position is analagous to what happened in the CAD/CAM market when people started to see the need to integrate CAD with CAM. The next five years is going to see this integration take place and at the same time the overall framework (analagous to CIM) will start to be worked out.

The system life cycle is now being defined to cover the full circle of operations, beyond even maintenance and including enhancement and renewal of systems.

Over the next two years standards will evolve and IBM will sanction the technology by launching its own repository based on DB2.

Loose coupling of methods to tools and vice-versa will be achieved using meta-methodologies that allow the tailoring of tools and methods to individual projects or to each DP shop.

There will remain two schools of thought, however:

- Close coupling of tools to methods will be favoured by one, e.g., the IEM—for proponents of close coupling, 'the methodology is a rule '.
- Loose coupling or the open-architecture approach, in which project management is more important than strict adherence to any one technological approach, will be the other school—here 'the methodology is a tool'.

The industry must find room for both schools of thought even if the rivals feel that the other is unnecessary or misguided.

B Opportunities Tied to the Life Cycle Approach

Vendors are recommended to consider three elements in their marketing mix. These correspond to three waves of innovation that are currently overlapping in the market-place:

- Training in methodologies and products
- Software tools addressed to individual life cycle functions
- Provision of complete systems, incorporating methods and tool kits

1. Training

The rare commodity in this field is the trainer himself. All countries researched stated that their software engineering schools were not producing enough staff able to train the next generation of system developers in the latest methods and tools. In other words technology was moving so fast that the production of the people required to propagate best practice was not happening.

This lack represents an opportunity for the consultancy community led by the methods houses that have marketed proprietary methodologies. Courses will increasingly need to be mounted to cover both the tools and the overall framework within which the tools and methods have to work.

Because adoption of software engineering principles requires that they are tailored to the individual DP environment, in-house courses will be more in demand than standard public courses. In-house courses are anyway higher margin services than public courses.

Other consulting assignments appropriate to the adoption of systems engineering must include services to assist in migration from one methodology to another or from one CASE technology to another.

2. Standalone Tools

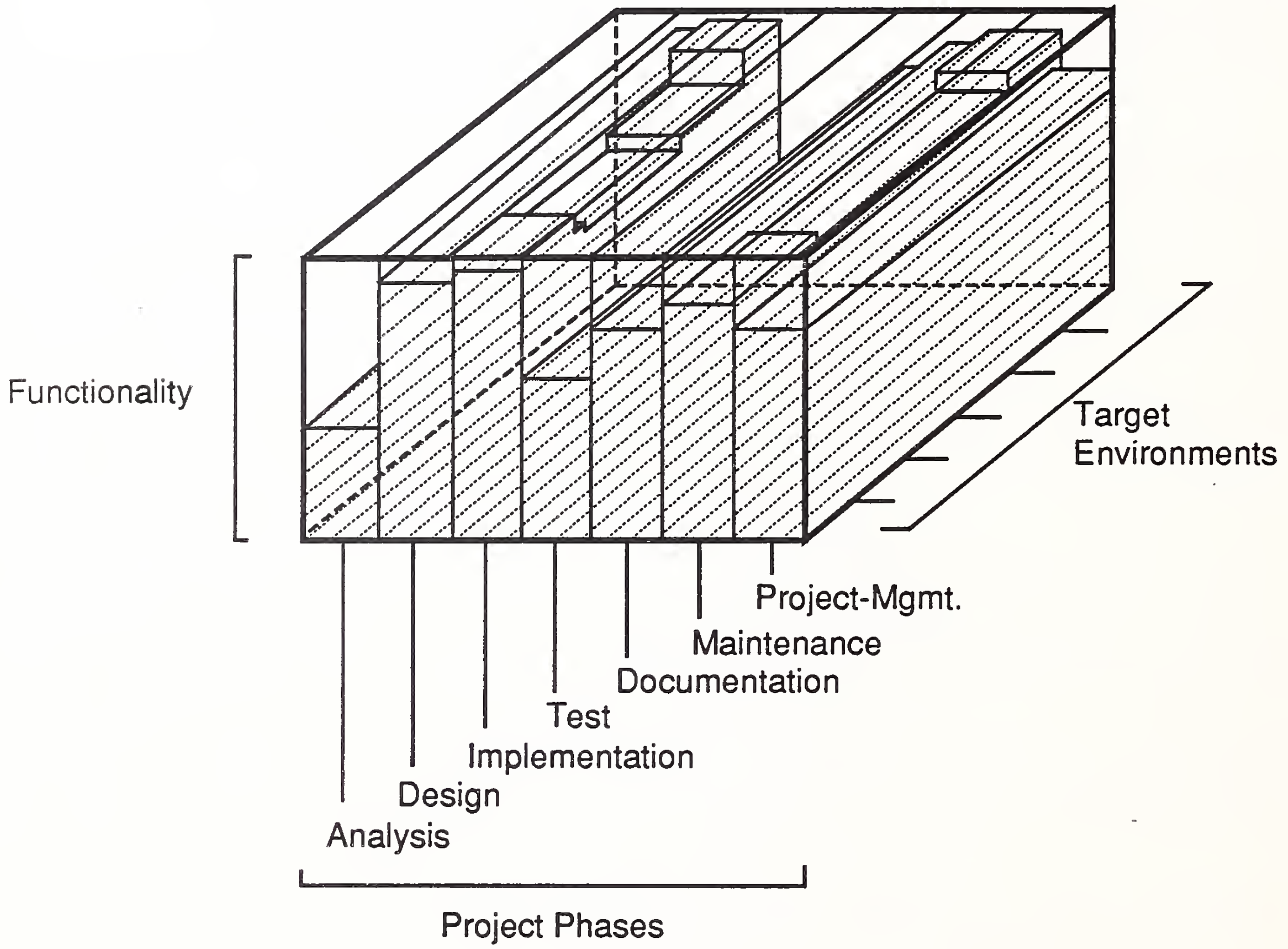
The immediate opportunity areas for tools are as tied to individual life cycle functions shown in the Cartesian diagram in Exhibit VIII-1:

- Strategic planning tools
- True prototyping and simulation tools
- Static testing tools
  - Reverse-engineering products
  - Data dictionary/repository interfaces
  - Conversion of existing tools to operate in other environments, e.g., with UNIX or on a C platform
  - Tool conversion for existing tools to operate with PCTE or IRDS interfaces
- Meta-tools for generating tools, tool sets and workbenches

Vendors should review their existing product portfolios to select areas suitable to in-house expertise and future plans. Vendors must decide whether they are going for niche markets or aim to be full-service vendors.

EXHIBIT VIII-1

OPPORTUNITIES FOR PRODUCT DEVELOPMENT USING THE BUILDING-BRICK CONCEPT OF THE LIFE CYCLE



This review will allow for selection of the correct product/service mix and the development of competitive functionality versus the competition.

3. Total Systems

These systems should be supplied as I-CASE systems or PCTE-based IPSEs, even if functionality is zero against certain areas of the life cycle—so long as the overall encyclopaedia and documentation management functions are covered adequately. The overall umbrella within which the systems must be sold comprises:

- Software products

- Turnkey systems

- Operations Management (FM)-type deals running in a network environment.

The overall life cycle must be considered when planning the evolution of vendor product catalogues, to take into account new platforms, new architectures and changing software technology.

During the 1990s, the software factory will become a reality, but different manufacturing methods will need to be applied. It is too early to specify the applicability of each:

- Production Line
- Group Technology
- FMS (Flexible Manufacturing Systems)

In this context the most applicable experience, which will reward study, is that of the CIM and EDA (Electronic Design Automation) vendors.

C Country Analysis

Vendors are looking to expand in various European country markets, and also searching for worldwide opportunities.

1. West Germany

This is a traditional market with the longest record in the use of sophisticated CASE tools. The START on-line system was built in the 1970s using MAESTRO at a cost of 110 man-years of effort. It has since been maintained with a maintenance staff of four, a tribute to the life cycle benefits of systems engineering.

The West German market is production oriented and therefore favours the bottom-up or middle-out approach. A good market entry point for this climate is to go in with an application generator. In-house methods are favoured; the user is looking for complementary services from product and consultancy houses.

With the very large number of small local software and professional services vendors (over 2,500), there is ample opportunity to work with local implementation agents for particular districts or regions in the country. Product can be sold to these suppliers on the basis of improved productivity and increased reusability of code. As these VARs expand and reach the stage of market awareness, these marketing considerations become important.

The vertical markets offering the best opportuniteis in the country are:

- Electronic and electrical industries
- Machine tools and heavy engineering
- Construction firms

National competition from Softlab, GEI and GPP must be expected.

2. France

The tools market is seen as still small and immature. The methodologies market is dominated by MERISE, with competition from AXIAL and EXPERT. CGI's PACBASE is present in large sites, and more potential for standalone tools exists in medium-sized users or subsidiaries of international companies.

Local presence in Paris will be necesssary for any serious market entry, with a joint venture to cover provincial users. Best opportunities for services are:

- Migration audits
- On-line CASE workshops via the NIS vendors

The most likely industries for market development are:

- Electronics
- Heavy engineering
- Financial services

3. U.K.

The U.K. is the fastest growing market in Europe at the present time—particularly with front-end tools based on the analyst workbench, such as Arthur Young's IEW; and at the back end with code and application generators, such as Cortex's CorVision and Pansophic's Telon. Inroads into large organisations have been made with the MAESTRO IPSE and James Martin Associates' IEF.

Companies that have changed over in a radical manner to the new technologies are very satisfied with the benefits obtained.

The results of the Alvey projects are now starting to come through in terms of commercial CASE products. A new government initiative with a consortium of systems houses is undertaking a project to develop a general methodology for the implementation of expert systems.

There is an increasing market for consultancy based on the correct product/service mix.

The best industry opportunities reside in the following sectors:

- Financial services
- Retail distribution
- Government

4. Italy

In Italy there are only a few relatively isolated islands of CASE competence in the large organisations. The penetration of structured methodologies is also low, due to the intensely individualistic nature of the Italian approach. The cottage industry flavour of early software development techniques will persist longer in this atmosphere than in any other of the major European country markets.

Vendors should appeal to the Italian user with sophisticated tools based on specific hardware platforms. For example, the local manufacturers—Honeywell Information Systems and Olivetti—provide standard UNIX and other micro-based hardware on which tools and services could be provided.

Marketing strategy should be geared to working through and with these well-known hardware suppliers.

The best industries as targets are likely to be:

- Financial services
- Government

5. Scandinavia

Norway is the home of object-oriented programming (OOP). Norway leads the way among the four Scandinavian nations, all of whom aim to be in the forefront of technology application as a matter of survival.

Scandinavia pursues its own R&D efforts in CASE technology with cross-country consortia of universities working on developing object-oriented CASE tools that will be commercialised through large professional services vendors or large users. Scandinavia is deeply penetrated with the application of structured methodologies; training will remain an important service sector.

There are fewer opportunities for large CASE systems due to the smaller number of large organisations. Opportunities exist for local vendors in the network services field with the application of CASE workshops on-line (a CASE workshop provides a range of CASE tools and services available to users in an on-demand mode for demonstration, trial periods and testing).

6. Benelux

The Netherlands is the largest market but again provides relatively few opportunities. However, there is a well-entrenched understanding of structured methodologies, both native and foreign. This means that there is a large market for analyst workbenches priced under $10,000. They must be sold into an open-architecture environment.

Back-end tools will be taken up more slowly and must be provided within an incremental approach to improving software engineering.

The awareness of structured methodologies and CASE tools in Belgium is extensive. The country, though small, offers a good short-term potential for products in both front- and back-end sectors. Vendors report high activity in 1988 and this indicates that the market for I-CASE systems in large multinational organisations is developing.

7. Rest of Europe

The Spanish market is large in potential but has a long way to go to catch up with other major countries in Europe. Users expressed ignorance of both structured methodologies and CASE technology. Therefore INPUT recommends that the primary requirement is for training and services. The Spanish mentality is well attuned to the one-stop shopping approach and similarly, as the market matures, products provided as turnkey systems will be well received.

Switzerland and Austria have a smaller number of opportunities but there is a thriving financial service community in Zurich with large IBM systems. 4GL and 5GL products are reported to be going strongly in this market. To be favoured, vendors will need to establish a balance between the incremental and the radical approaches. Technology migration audits are a useful consultancy opportunity.

8. Outside Europe

The U.S.A. is more highly developed as a user of CASE tools than any of the European markets, but tools have been implemented mainly in standalone mode as aids to the professional developer. There is a growing awareness of the need to structure the application of CASE within the framework of proven methodologies. In consequence a natural slowing of the market can be observed, with year-on-year growth having slowed to about 60%, compared with a previous rate nearer 100%.

IBM has announced the intention to develop its own repository; this means the market has entered the FUD (Fear, Uncertainty and Doubt) period. The market will not escape FUD until the announcement is followed by actual launch.

The U.S. market has reached the limit of what can be done with tools alone and is now ready for the IPSE or I-CASE approach. However, vendors must realise that in each sales situation the business case must be made. Market entry from outside the U.S.A. should be made with the assistance of professional marketing advisers who know the sales and marketing requirements of this very tough market—local management, sales employees with track records, acquisition routes etc. For further information on the CASE market in the United States, readers are refered to a separate INPUT report, *CASE Markets 1988-1993*.

The Japanese market has an intense need for software professionals. The education system has not produced the numbers required, and a very large gap is expected itself over the next five years. This gap favours CASE. There will be opportunities for service houses who can justify the initial market entry costs and traditional long lead times in Japan. Joint ventures with local suppliers are recommended.

A

Appendix: Definitions

A

Appendix: Definitions

The following are definitions in alphabetical order of terms used in software engineering:

Aid - A computer based-technique.

Automated - Computer based.

Back-End Tools - Software development tools (see Tools) used in the production, testing and maintenance stages of the life cycle. Now also known as Lower CASE tools.

Data Analysis - A formal method of examining data structures to ensure their freedom from embedded information (such as repeating groups, inter-item dependency, etc.) which would increase maintenance difficulties.

Definition or Proposal Stage - Completes when the customer for the system agrees an overall target specification.

Design Test Plan - Indicates the types of tests considered at each level of implementation.

Development Library - Holding different versions of software and controlling their release and use.

Diagrammatic Representation of Data Flows - Can be used in entity, logical, and physical modeling. Techniques include Structured Analysis and Design Techniques (SADT), Yourdon, Warnier-Orr and Gane and Sarson.

Enhancement - Meet revised requirements.

Entity Modeling - Identifying the source and destination of data, data in the system, and its transformation and storage.

Front-End Tools - Tools used in the analysis and design stages of the life cycle. Now also called Upper CASE tools.

I-CASE (for Integrated CASE) Tools or Systems - Tools or systems that address/purport to address the total system life cycle.

Integration Testing - Testing of linked modules or programs as a unit.

Logical Model - States what the system is required to do, expressed in data flow and data relationship diagrams, in such a way as to be readily understood by non-technical users. The logical model may then be verified in conjunction with the user and against the stated used requirements.

Lower CASE Tools - See Back-End Tools.

Methodology - A methodology is considered to be a formal collection of explicit or referenced principles, policies, methods, procedures, practices, and definitions detailing how something is to be accomplished. Essentially, it provides a route map for project personnel-related to each key project development aspects.

Maintenance - Correction of the operational system.

Physical Model - Maps the logical model onto the hardware and software. It is created by the technical system designer. Diagramming techniques may be used to derive a structured design which can be verified.

Portability - Capacity of the software product to be moved to other hardware.

Project Management - Sets out how projects will be accomplished and specifies the interfaces with other organisational groups such as users. Techniques can include project estimating, planning and control of resources, monitoring progress, authorisations, use of other methodologies and development techniques. A policy statement for the project or, indeed all projects, may be one technique; use of a Project Manual as a reference and a record of decisions can be another.

Project Stage Documentation and Change Control - Ensures that each stage of a project has a formally documented end-product that is subject to formal review and acceptance sign-off and from then on subject to formal change control to ensure that all levels of documentation fully incorporate approved changes; also ensures configuration control of product build-state.

Project Stages - Are of necessity somewhat arbitrary but are considered to complete as follows:

- Definition or proposal completes when the customer for the system agrees on overall target specification.
- System specification and design are complete when an initial design document identifies and to some extent defines the six to twenty most significant items in terms of design effort.
- Software design and development are completed when operational use of the system begins. In certain circumstances an earlier completion is acceptable where:
  - Extensive parallel running is required for customer acceptance.
  - The system is put "on ice" awaiting an external event.

In these cases it is usually identifiable that a maintenance team is in post and therefore the project is complete.

Prototyping - Involves demonstrating, confirming, or improving the system specification by building a subset or model of the full system.

Quality Assurance (QA) - Applied to specifications and software end products. Also QA may be applied to the means by which such products are produced, i.e. project organisation, plans, communications control procedures, standards, reviews, or walkthroughs.

Software Construction and Documentation Methodology - Using techniques such as:

- Structured program design through use of data structure analysis, e.g. Warnier-Orr, Jackson.
- Structued coding to implement programs through use of sequence selection and iteration language constructs.

- Top down software build using a module hierarchy beginning at the root level then progressing through more detailed levels.

- Development library holding different versions of software and controlling their release and use.

- Software generators providing automatic code generation for screen formats, reports and programs.

- Walkthroughs of code in an informal manner to critique all related aspects against the programming concensus.

Software Design and Development - Implementation of the software system through detailed design, code, and test.

Software Development Cycle - Covers only system design, software engineering, coding, testing, and integration.

Software Engineering (SE) - Encompasses all technical and management activities undertaken during the development of software systems from initial requirements defintion through implementation to customer acceptance and beyond into systems maintenance.

Software Engineers - Are those people involved in the production of software at either a management or technical level and thus include specifiers, designers, programmers, quality staff, document action specialists and managers.

Software Engineering Tools - Are automated techniques which support some component part of a methodology. Tools are available to support all types of methodologies; including:

- Project management.
- Stage documentation and change control.
- Quality assurance.
- Systems requirements.
- Systems design.
- Software construction and documentation.
- Software and system testing.

When discussing the potential market for software engineering tools, only substantial tools which might cost in excess of £5,000 for medium to large systems, have been considered.

Software Generators - Are created by a project team using the SE disiciplines. The project life cycle is split into project stages.

Software Life Cycle - Used to cover all aspects of product definition, development, installation, and maintenance. This is in contrast to software development cycle.

Specification - A software system requirement.

Specification Team Walkthroughs - Review and confirm in an informal manner the system specification.

Structured Coding - Implementation of programs through use of sequence selection and iteration language constructs.

Structured Program Derived Test Data - That which uses the program structure to ensure each path through the developed code is exercised.

Structured Program Design - Through use of data structure analysis, e.g. Warnier-Orr, Jackson.

System Design - Initial translation of the specification into the main components for implementation.

System Requirements - Producing the specification using techniques such as entity modeling, prototyping, or specification team walkthroughs.

Systems Specification and Design - Completes when an initial design document identifies and to some extent defines the six to twenty most significant items in terms of design effort.

Techniques - Component parts of a methodology.

Test Plans - Setting out test objectives, timescales, inputs, and expected results.

Test Team - Responsible for independently conducting tests and formally returning or accepting delivered programs.

Testing Methodology - The collection of techniques for testing including test plans, top down testing, test team, and integration testing.

Tool - An automated technique for software development incorporated in and marketed as a software product.

Top-Down Program Testing - To progressively exercise more and more of the programm as it is built to ensure all module paths and linkages are tested.

Top-Down Software Build - Uses a module hierarchy beginning at the root level, then progressing through more-detailed levels.

Top-Down System Testing - Used in a similar incremental way to ensure consistency between programs regarding use of files.

Unit Testing - Carried out by the programmer responsible for developing code.

Upper CASE tools - See Front-End Tools.

Visibility - Capability for outsiders to assess project progress.

Walkthroughs of Code - To critique, in an informal manner, all related aspects against the programming concensus.

B

Appendix: Analysis of Research Sample

B

Appendix: Analysis of Research Sample

In-depth interviews (all face-to-face) were conducted amongst vendors of proprietary methodologies and vendors of systems development productivity tools (i.e., workbenches, applications generators, advanced programming languages, and integrated project support environments [IPSEs]).

Exhibit B-1 shows the analysis by country of operation of the vendor organisations interviewed.

EXHIBIT B-1

VENDOR INTERVIEW PROGRAMME

Country	Number of Interviews
France	5
West Germany	6
United Kingdom	14
Italy	1
Scandinavia	1
Benelux	2
Multinational	6
Total	35

Telephone interviews were also conducted amongst a wide cross section of user organisations in France, West Germany, Italy, Belgium, the Netherlands, Sweden, Finland, Norway, and the United Kingdom.

This user research was conducted as part of INPUT's bi-annual survey of Western European Software and Services markets which focused on usage and attitude towards systems development productivity tools and techniques as well as software product pricing and support.

Exhibit B-2 shows the analysis of user organisations interviewed.

EXHIBIT B-2

USER INTERVIEWS SAMPLING FRAME IS MANAGEMENT

Number of Interviews	Industry Sector	Number of Interviews	Country Market
14	Discrete Manufacturing	21	France
12	Process Manufacturing	19	West Germany
4	Retail Distribution	20	United Kingdom
17	Wholesale Distribution	10	Italy
5	Transportation	10	Scandinavia
10	Utilities	10	Benelux
22	Banking and Finance	10	Spain
1	Insurance	-	
2	National Government	-	
7	Local Government		
6	Other		
100	All Sectors	100	All Countries

In addition, a number of key software development executives in user organisations were interviewed face-to-face in order to clarify INPUT's understanding of key trends and issues.

C

Appendix: Vendor Questionnaire

C

Appendix: Vendor Questionnaire

Study Title: *Towards the 5th Generation—Tools & Rules* Study Code: SFGE

Type of Interview □ SP Coy.
□ SSH/SSII □ Tele.
□ User □ On-Site
□ Other □ Mail Date: _____/_____/88

Respondent ______________________ Function/Title ______________________

Company Name ______________________ Division ______________________

Address ______________________ No. Telephone (____) ______________

____

____

Staff Nos. (W-wide) ____________ (Divn.) ____________ (. . . . . .) ____________

(Europe) ____________ (Home Market) ____________

All Sectors (1987...) / /£/LIT/DM/FF ____________ ($ ____________)

Revs.-Europe (198...) for . . . . ____________ ($ ____________)

. . . . .(198...) for . . . . ____________ ($ ____________)

Can we have a copy of your latest Annual Report?

Q.1 Do you market 4GL Development Tools/Environments? How important are they in revenue terms?

% of European revenues ____________ □ Not at all

□ Minor Earner

No. ______ ____________ □ Major Revenue Earner

- Population Installed ____________
- Names of Products ____________

____

Q.2 Do you intend to continue to offer them?

____

Q.3 How does this fit in with your principal strategic axes?

____

____

Q.4 Do you market any Software Implementation Methodologies, or any tools to assist in their use?

- Population Installed

1. ____________ ____________
2. ____________ ____________
3. ____________ ____________

Q.5 Do you market any CASE toolsets or tools, or 5th Generation Development products or tools?

- Population Installed

1. ____________ ____________
2. ____________ ____________
3. ____________ ____________

Q.6a Which manufacturers' equipment is supported? □ Own only

□ IBM □ DEC □ ICL □ Others ____________

□ ____________

Issues

Portability ____________

Integration ____________

Q.6b Which methodologies are supported?

- ❑ SSADM (LDMS)
- ❑ Core
- ❑ JDA
- ❑ MASCOT 3
- ❑ Yourdon
- ❑ Gane & Sarson
- ❑ Own only
- ❑ Others' ____________
- ❑ ____________
- ❑ In-House

Issues

Flexibility ____________________________

Integration ____________________________

Q.6c Which suppliers' database products are supported?

- ❑ Oracle
- ❑ IBM (DB2)
- ❑ Relational Technology (Ingres)
- ❑ DEC (RDB,RMS)
- ❑ Cullinet (IDMS)
- ❑ ICL
- ❑ Own only
- ❑ Software AG (ADABASE)
- ❑ Others ____________
- ❑ ____________

Issues

Migration ____________________________

Integration ____________________________

Q.7 Do you distinguish between the 4th Generation of software development and the 5th? And, if so, how?

____

Q.8 Which phases of the Systems Development Life Cycle do your products support?

Phase	Y/N	Comment on How
Project Management		
Business Analysis		
Systems Design		
Code Generation		
Testing		
Documentation		
Quality Assurance		
Maintenance/Support		
Enhancement/Migration		
Other ____________		

Q.9 Which ancillary services are also offered?

Bundled/Unbundled

Application Programming	____________	❑	❑
Professional Services	____________	❑	❑
Education/Training	____________	❑	❑
Contract Personnel	____________	❑	❑
Other	____________	❑	❑

Q.10 Please comment on the importance of the following aspects/features

Data Dictionary, active ____________________

Standard Query Language ____________________

Distributed Database ____________________

Interface mechanisms ____________________

Other __________ ____________________

(e.g., parallel processing, neural networks)

Q.11 Do you favour an 'incremental prototyping' approach or a strictly hierarchical one? Or can you support both?

____

Q.12 Which benefits are you stressing to your prospects/customers will follow from the use of these products? (Please rank in order of importance, from 1 down)

Rank	Comments	Benefit
		Improved productivity
		Improved quality of software
		Improved flexibility of design
		Easier maintenance
		Easier migration
		Preservation of investment
		Use by lower calibre staff
		Other __________

Marketing

Q.13 Which group of users are you targeting? Please rank in order of importance.

Rank	Type of User	Comment
	Information Centre user	
	In-house development staff	
	Systems House/VARs	
	Other __________	

Q.14 What are the 'drivers' and 'inhibitors' to this market sector?

Drivers

____

Inhibitors

____

Q.15 What is your estimate of the size and growth of the sector?

In your domestic market ____________________ pa. ________________ % AAGR

Rest of Europe ____________________________ pa. ________________ % AAGR
(N.B. Use value (local currency) or shipments)

Q.16 Who are your major competitors?

Domestic market	Rest of Europe
1. ______________________	______________________
2. ______________________	______________________
3. ______________________	______________________

Q.17 How do you market your Advanced Software Development Products and Methods? (Please detail and comment)

Direct sales force ______________________________ ____% of sales force of ____
OEMs/VARs _________________________________ ________ no.
Other ____

____

Q.18a What size of company are you serving with these products/methods?

	% of customer base
❑ Large, i.e., with >5,000 staff (>200 IT staff)	__________%
❑ Medium, i.e., with 1,000 to 4,999 staff (50 to 200 IT staff)	__________
❑ Small, i.e., with < 1,000 staff (<50 IT staff)	__________

Q.18b And where do you think the potential is?

____

Q.19 Is there any industry bias in the market potential?

____

____

Q.20 How does your market vary across the different countries of Western Europe?

	Europ. Revs. %
W. Germany ______________________________________	____________
France ____	____________

U.K. ______________________________ __________

Italy ______________________________ __________

Benelux ______________________________ __________

Rest ______________________________ __________

Q.21 Are there any other important aspects not covered so far?

____

THANK YOU FOR YOUR TIME

D

Appendix: User Questionnaire

D

Appendix: User Questionnaire

Software Engineering and Applications Software (C)

DP Management

February 1988

A
Background Information

Respondent Name: ____

Title: ____

Name of Company: ____

Address of Establishment: ____
(including country) ____
____

Telephone No. (in full): ____

QA May I check on the total number of full-time employees in your establishment?

500-999	1
1,000 +	2

QB What is the principal activity of your organisation?

Manufacturing - Discrete	1
Manufacturing - Process	2
Distribution - Retail	3

Distribution - Wholesale	4
Transportation	5
Utilities	6
Banking and Finance	7
Insurance	8
Government - National	9
Government - Local	0
Other ________________	X

Introduction

Good morning, good afternoon, I am telephoning from INPUT, the international planning services company. We are conducting a survey about the major issues in the data processing industry. Your opinion would be very valuable. All information you give will of course be absolutely confidential.

QC What are the main types of data processing equipment that you have at your company (this establishment)?

Probe whether mainframe/mini, etc.

Mainframe □
Mini □
Workstation/PCs □
Other ____________ □

B

Software Engineering

Q.1a Over the period 1987-1988, has your application backlog increased, decreased or remained the same?

Increased □
Decreased □
Remained the Same □

Q.1b *If Increased or Remained the Same,* what are the major constraints on your ability to reduce the applications backlog?

1. ______________________________
2. ______________________________
3. ______________________________

Q.1c *If Decreased,* what are the major factors enabling you to reduce the applications backlog?

1. ______________________________
2. ______________________________
3. ______________________________

Q.2 Could you please estimate the approximate percentage of the total time spent by applications development staff on the following activities:

Activity	Percentage
Enhancement of Existing Systems	________
Maintenance of Existing Systems	________
Development of New Systems	________
TOTAL	100%

Q.3 Turning specifically to Software Engineering and computer-aided software engineering technology:

a) Do you use or are you planning to use a fully integrated CASE system in your organisation?

N.B. A suite of software products or system that supports all phases of the development life cycle, from concept to code.

E.g. In the U.K., an Integrated Project Support Environment; Software Produktions Umgebung (SPU) in Germany and Atelier de Genie Logiciel (AGL) in France.

If using/planning to use/evaluating:

b) What were/are your organisation's major objectives in deciding to invest in an advanced CASE system?

1. ______________________________
2. ______________________________
3. ______________________________

c) What, if any, do/did you perceive as being the major difficulties that you faced when evaluating CASE products?

1. ______________________________
2. ______________________________
3. ______________________________

d) What were the major problems (if appropriate) that you encountered when implementing CASE technology in your organisation?

Prompt:
- Training
- Use of Structured Methods

1. ______________________________
2. ______________________________
3. ______________________________

If not:

e) Why do you think that the use of advanced CASE systems is an inappropriate strategy for your organisation?

1. ______________________________
2. ______________________________
3. ______________________________

Prompt:
- Cost
- Inappropriate to Stage of DP Development

Q.4 Are you using/planning to use any structured methodologies in systems development?

If Yes - Are you using proprietary and /or in-house methodologies?

- Proprietary Methodology □
  e.g., Jackson, Yourdon, Merise, SSADM, Information Engineering, etc.
- In-House Developed □
- Both □

If No - Why do you feel that the use of structured systems development methodologies is an inappropriate strategy for your organisation?

1. ______________________________
2. ______________________________
3. ______________________________

Q.5 Please rate the importance of the following potential actions that may be implemented as part of a plan to improve the efficiency, effectiveness and quality of software development?

Factor	Rating (0-10)
Increased End User Involvement	□
Training - Project Management Skills	□
- Methodology	□
- Tools	□
Increased Awareness of Business Problems	□
Increased Commitment to Quality	□
Change in Organisational Style and Culture	□